A Variability-Aware Design Approach to the Data Analysis Modeling Process

by

MariaCristina Vale Tavares

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2018

AUTHOR'S DECLARATION

Abstract

The massive amount of current data has led to many different forms of data analysis processes that aim to explore this data to uncover valuable insights such as trends, anomalies and patterns. These processes support decision makers in their analysis of varied and changing data ranging from financial transactions to customer interactions and social network postings. These data analysis processes use a wide variety of methods, including machine learning, in several domains such as business, finance, health and smart cities.

Several data analysis processes have been proposed by academia and industry, including CRISP-DM and SEMMA, to describe the phases that data analysis experts go through when solving their problems. Specifically, CRISP-DM has modeling as one of its phases, which involves selecting a modeling technique, generating a test design, building a model, and assessing the model. However, automating these data analysis modeling processes faces numerous challenges, from a software engineering perspective. First, software users expect increased flexibility from the software as to the possible variations in techniques, types of data, and parameter settings. The software is required to accommodate complex usage and deployment variations, which are difficult for non-experts. Second, variability in functionality or quality attributes increases the complexity of these systems and makes them harder to design and implement. There is a lack of a framework design that takes variability into account. Third, the lack of a more comprehensive analysis of variability makes it difficult to evaluate opportunities for automating data analysis modeling.

This thesis proposes a variability-aware design approach to the data analysis modeling process. The approach involves: (i) the assessment of the variabilities inherent in CRISP-DM data analysis modeling and the provision of feature models that represent these variabilities; (ii) the definition of a preliminary framework design that captures the identified variabilities; and (iii) evaluation of the framework design in terms of possibilities of automation. Overall, this work presents, to the best of our knowledge, the first approach based on variability assessment to design data modeling process such as CRISP-DM. The approach advances the state of the art by offering a variability-aware design a solution that can enhance system flexibility and a novel software design framework to support data analysis modeling.

Acknowledgements

I would like to thank Professor Paulo Alencar for his support and guidance. I’m very grateful to Professors Donald Cowan and Daniel Berry for serving as my co-supervisors. I also thank Professors Jo Atlee and Gladimir Baranoski for agreeing to read my thesis and giving me valuable feedback.

Table of Contents

AUTHOR'S DECLARATION ii
Abstract iii
Acknowledgements v
Table of Contents vi
List of Tables viii
List of Figures ix
List of Abbreviations x
Chapter 1 Introduction 1
1.1 Background 2
1.1.1 Data Analysis Processes 2
1.1.2 Variability Modeling and Analysis 3
1.1.3 Object-Oriented Frameworks 4
1.2 Problem 5
1.3 Proposed Approach 6
1.4 Contributions 7
1.5 Thesis Outline 8
Chapter 2 Related Work 9
2.1 Data Analysis Processes 9
2.1.1 Existing Data Analysis Processes 10
2.1.2 CRISP-DM 12
2.1.3 CRISP-DM Modeling Phase 14
2.1.4 CRISP-DM Extension Approaches 16
2.2 Variability Modeling and Analysis 17
2.2.1 Feature Models 17
2.2.2 UML Modeling 20
2.3 Object-Oriented Frameworks 22
Chapter 3 A Variability-Aware Data Analysis Modeling Process 23
3.1 Variability Assessment 23

3.2 Framework Design Model 39
3.3 Evaluation of Automation Opportunities 43
3.3.1 Select Modeling Technique 45
3.3.2 Generate Test Design 46
3.3.3 Build Model 47
3.3.4 Assess Model 48
3.4 Discussion 49
Chapter 4 Conclusions and Future Work 51
4.1 Conclusions 51
4.2 Future work 51
References 53
Appendix A CRISP-DM Variability Assessment 61
Index 76

List of Tables

Table 1 - CRISP-DM vs. other data analysis process approaches. 11

Table 2 - Feature diagram notation. 19

Table 3 - Comparison between SPSS Modeler CRISP-DM and framework design in the Selecting Modeling Techniques task. 45

Table 4 - Comparison between SPSS Modeler CRISP-DM and framework design in the Generate Test Design task. 46

Table 5 - Comparison between SPSS Modeler CRISP-DM and framework design in the Build Model task. 47

Table 6: Comparison between SPSS Modeler – CRISP-DM and framework design in the Assess Model task. 48

List of Figures

Figure 1 – A variability-aware design approach to the data analysis modeling process. 7

Figure 2 - Evolution and relationship among data analysis process models (from [43]). 10

Figure 3 - CRISP-DM data analysis phases (from [9])... 12

Figure 4 - CRISP-DM data analysis modeling generic tasks (from [9]). 15

Figure 5 - Example of a feature diagram. 19

Figure 6 - Example of a feature diagram from [53]. 20

Figure 7 - A variability-aware design approach: feature diagrams... 24

Figure 8 - Feature diagram for the CRISP-DM modeling phase. 26

Figure 9 - Feature diagram for feature Select Modeling Technique....................................... 28

Figure 10 - Feature diagram for feature Test Design... 32

Figure 11 - Feature diagram for feature Build Model... 34

Figure 12 - Feature diagram for feature Assess Model... 37

Figure 13 - A variability-aware design approach: framework design. 39

Figure 14 - Framework design for CRISP-DM modeling phase. 41

Figure 15 - A variability-aware design approach: automation opportunities. 43

List of Abbreviations

CRISP-DM	Cross-Industry Standard Process for Data Mining
FODA	Feature-Oriented Domain Analysis
KDD	Knowledge Discovery in Databases
NLP	Natural Language Processing
SEMMA	Sample, Explore, Modify, Model, and Assess
SPSS	Statistical Package for the Social Sciences
UML	Unified Modeling Language

Chapter 1
Introduction

The massive amount of current data has led to many different forms of data analysis processes that aim to explore this data to uncover valuable insights such as trends, anomalies and patterns. These processes support decision makers in their analysis of varied and changing data ranging from financial transactions to customer interactions and social network postings. These data analysis processes use a wide variety of methods, including machine learning and statistical data analysis, in several domains such as business, finance, health and smart cities [1], [2], [3]. In general, data analysis approaches have the potential to advance significantly the development of descriptive, diagnostic and predictive data analysis applications.

The process of data analysis involves phases such as data understanding, data preparation, modeling, evaluation and deployment. Specifically, the data modeling phase uses specific data analysis models to generate results in various situations. This phase in general has tasks such as select modeling technique, generate test design, build model and assess model. Each of these tasks has numerous variations. For example, the selection of specific models to be applied depends on factors such as the application domain, business goal, suitability of the model type, and data assumptions and available data types.

The high-level goal of this work is to make it easier to perform the data analysis modeling process. To accomplish this goal there is a need to increase the extent to which this process can be automated.

This thesis assumes that the analysis of the variabilities associated with the data modeling process can lead to capturing the variations in this process in a more comprehensive way using feature models and to producing a flexible software framework design. This design can help identify opportunities for automation that go beyond the automation support provided by existing tools. In this context, the thesis offers a variability-aware design approach to the data analysis modeling process.

This chapter describes the background and motivation, the challenges, the proposed approach, and the contributions of this research to the state of the art in the area of data analysis processes.

1.1 Background

1.1.1 Data Analysis Processes

Data analysis is the process of applying approaches and computer systems to examine in-depth data, using tools, techniques, and methods, so that meaningful information can be identified to make decisions or solve problems [4], [5], [6]. A simplified view of a data analysis project is that it comprises four main phases including: preparation, preprocessing, analysis, and post-processing [5].

Hardware and database technology progress has allowed the collection, storage, and access of all kinds of digital data at a growing and rapid scale [5], [7], and this data has been increasingly used in data analysis processes. Sources of data – banking machines, Internet portals, mobile devices, slot machines – are diverse and have different levels of reliability and complexity. This data comes from several domains, including industry, business, medicine, biological processes, and is based on different types, such as numerical, text, structured, and image data [5], [3], [6]. In addition, the availability of such data has led to new approaches to data understanding and analysis that can help turning data into useful knowledge to improve decision making, create new businesses, and reduce costs[1], [3], [7], [8]. Because manual data analysis of such huge volumes of data is not only impractical and in some cases almost an impossible task [7], there is an increasing need to automate the data analysis process.

Several data analysis processes have been proposed by industry and academia to describe the phases that data analysis experts should go through when solving their problems, including CRISP-DM (Cross-Industry Standard Process for Data Mining) and SEMMA (Sample, Explore, Modify, Model, and Assess) [9], [10, 11], [12]. These data analysis processes are widely used in industry to support current data analysis efforts. Specifically, CRISP-DM has modeling as one of its phases, which involves tasks such as selecting a modeling technique, generating a test design, building a model, and assessing the model.

1.1.2 Variability Modeling and Analysis

Variability is “the ability of a software system or artifact to be efficiently extended, changed, customized or configured” [13]. It allows a software system to be reusable and customized in a variety of scenarios and domains, as well as in its multiple versions [14], [15], [16], [17], [18]. In earlier stages of a software system design, some requirements options can be defined in advance and new versions of the system can be designed to incorporate some of these options [18], [19], [20]. In these terms, variability analysis can support the planning for future adjustments, adaptations and changes. Over the system software life cycle, the variability analysis includes the identification of the software system parts that vary and the range of options for these variations. A variation point refers to a representation of an item that may vary, such as “technique” or “payment method,” and a variant is a representation of a single option that a variation point may assume (i.e., for the variation point “technique,” the variants could be “decision-tree” or “K-means,” and for “payment method,” the variants can be “credit card” or “cash”). The relationship among variation points and their variants establish dependencies that can be mandatory or optional [21], [22].

A major impact of the variability analysis on the software engineering area is that it provides adaptability, flexibility, and reusability for the software artifact throughout the system lifecycle [14], [23]. The exploration of the commonalities and variabilities of software products helps to improve the development efficiency and enables the configuration of software system (behavior, structure and process) [14], [17], [24], [25].

Variability facilitates the adaptation of a software system to changes and allows reusability of software components [17]. Variability has been applied to many different scenarios of software systems, including dynamic selection of features, configuration tools, or self-adaptive systems [17]. The focus of variability analysis is to identify the components that are candidates to be extended, customized or changed, according to the context or domain.

Studies on variability in software systems have been conducted and many variations of modeling approaches have been developed to implement variability techniques [24], [17], [26]. There are numerous types of variability models, including feature models [24], decision modeling [24], and models based on UML [21], [22], [27]. Feature models, for example, are used to understand, define, and communicate the variability of a software system through the representation of a logical relationship between variation points and variants [21], [13], [28]. Decision modeling focuses on identifying possible decisions for domain-related questions to be used in the product configuration [24]. Variability models based on UML are used to model a family of software systems that can support some common functionality and some variable functionality [27]. Overall, variabilities can be assessed in many ways, one of which is by analyzing nouns and verbs in requirements system documents [51].

Specific software system behavior, which may involve change, configuration, or extension, is enabled by means of the selection of variation points and their variants. This ability makes it possible to customize a system to specific circumstances, and this may include component customization, feature selection, or adaptation at runtime, and system configuration. Further, the variability model is flexible and reusable in the sense that additional variants can be included with respect to a variation point, which implies that some decisions in the development process can be delayed [23]; rather than deciding beforehand, a customized solution can be configured later according to specific system requirements.

1.1.3 Object-Oriented Frameworks

Researchers and practitioners have been studying software reuse for decades and several techniques, tools, and applications for this purpose have emerged over the years [29]. The origin of the framework concept dates from mid-80s when it was defined as “a large structure that can be reused as a whole for the construction of a new system” [30] for a particular domain. A framework can also be seen as an object-oriented abstract design [31] that refers to “a generic architecture that provides an extensible template for applications within a domain” [15], by means of which developers express reusable designs for a software system [32, 33], [34]. Alternatively, it can be defined as “a set of classes that embodies an abstract design for solutions to a family of related problems" [31]. The role for object-oriented framework is important for the development of open systems [29] as well as in a wide range of domains. Fire-alarm systems, gateway billing systems, robot control, business-oriented applications, and speech recognition are some examples of reported framework applications [30], [35].

Software frameworks lead to several benefits, including higher reuse and productivity and improved quality in the software system development process [29], [30], [36], [37], [38]. Higher productivity comes from the fact that developers have a core system and possible extensions from which the system can be built. Improved quality results from the reuse of pre-defined and tested components and their variations. The reuse of design patterns, components, and architectures is the foundation of the object-oriented reuse technique [39], [34], [38], enabling the customization and adaptation of applications within a specific domain [40], [33]. Customization may involve invoking operations of components, exchanging data, or even handling errors [34]. To tailor a generic architecture for a specific situation or problem in a given domain, frameworks provide a flexible and reusable design foundation along with its elements, relationships, and extension points that can be modified, specialized, and extended to build a new application [35], [41], [30], [33], [42].

1.2 Problem

The software automation of data analysis modeling processes from a software engineering perspective faces numerous challenges. First, software users expect increased flexibility from the software in terms of the possible variations regarding different techniques, types of data, and parameter settings. The software is required to accommodate complex usage and deployment variations, which are difficult for non-experts. Second, variability in functionality or quality attributes increases the complexity in the design space of these systems and makes them harder to design and implement. There is a lack of a framework design that takes variability into account to support the design process. Third, the lack of a more comprehensive analysis of the variability design space makes it difficult to evaluate opportunities for automating the data analysis modeling process.

The thesis addresses three main research questions:

(i) What are the variabilities related to the CRISP-DM data analysis modeling process and how can these variabilities be represented using feature models?

(ii) What framework design can capture the variabilities in the CRISP-DM data analysis modeling phase?

(iii) What are the opportunities for automation of the CRISP-DM data analysis modeling phase that go beyond the support provided by existing tools?

1.3 Proposed Approach

This thesis proposes a variability-aware design approach to the data analysis modeling process. The approach involves (i) the assessment of the variabilities inherent in the CRISP-DM data analysis modeling phase and the provision of feature models that represent these variabilities; (ii) the definition of a preliminary framework design that captures the identified variabilities; and (iii) the evaluation of the proposed framework design in terms of the possibilities for process automation.

The assessment of the variabilities is based on the data analysis modeling phase of the CRISP-DM process model, the de-facto standard methodology for data analysis projects [9], and highlights the complex dependencies among variation points and their variants. The preliminary framework design aims at providing a flexible design model that can be used as a basis for implementation purposes. The evaluation of the framework design in terms of possibilities for automation can lead to enhancements in the level of automation supported by existing tools.

The proposed approach is illustrated in Figure 1. According to this figure, first, a variability assessment is conducted using as sources the CRISP-DM documentation [9], [11] and several articles that support the classification of the techniques that can be applied in the data analysis modeling process and, based on the variability assessment, feature diagrams are developed to represent the variabilities present in this process. Second, the feature diagrams are used to create a preliminary framework design for the CRISP-DM modeling phase. Third, the opportunities for automation of the CRISP-DM data analysis modeling phase that go beyond the support provided by existing tools are evaluated.

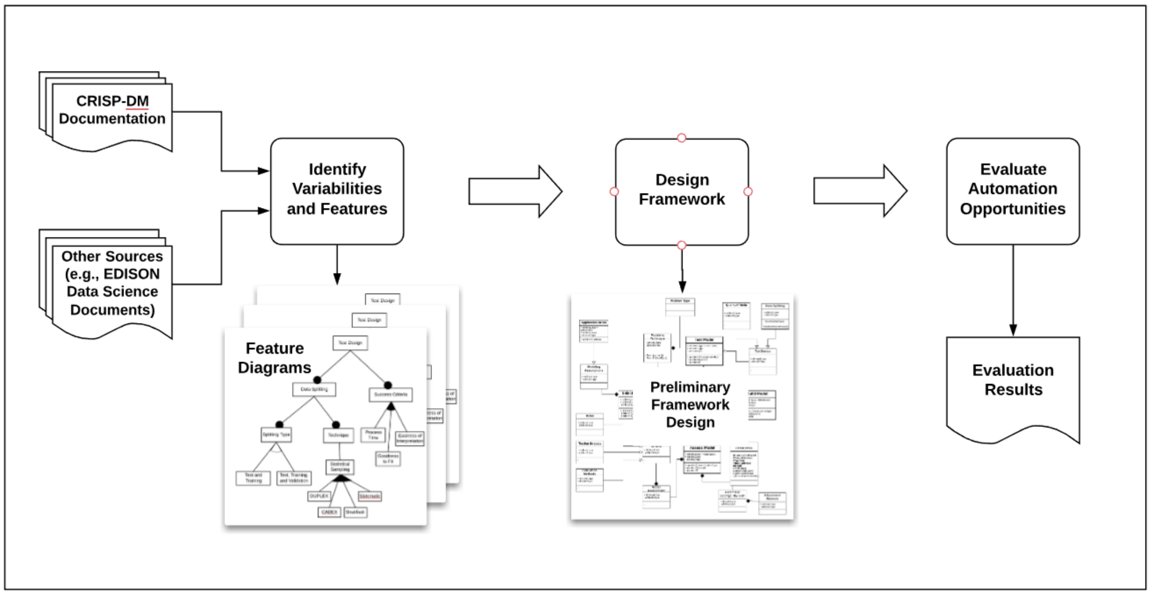

CRISP-DM Documentation
Other Sources (e.g., EDISON Data Science Documents)
Identify Variabilities and Features
Feature Diagrams
Design Framework
Preliminary Framework Design
Evaluate Automation Opportunities
Evaluation Results

Figure 1 – A variability-aware design approach to the data analysis modeling process.

1.4 Contributions

The thesis statement of this research work is the following:

> **Assessing the variabilities in the CRISP-DM data analysis modeling process can contribute to the development of feature models and a design framework that indicates new opportunities for automation for the CRISP-DM modeling phase**.

The contributions of the thesis include:

(i) Variability assessment of the CRISP-DM data analysis modeling process, which provides variations related to its four generic tasks, namely Select Modeling Technique, Generate Test Design, Build Model and Assess Model, and the feature diagrams that represent these variations;

(ii) A preliminary framework design that is developed based on the four generic tasks of the variations identified in the variability assessment; and

(iii) An evaluation of the possibilities for automating the data analysis modeling process for each of the four generic tasks.

Overall, this work presents, to the best of our knowledge, the first approach based on variability assessment to design data analysis modeling processes such as CRISP-DM. The approach advances the state of the art by offering a variability-aware design solution that can (i) enhance the flexibility of systems for data analysis modeling; and (ii) potentially lead to novel software frameworks that improve significantly the level of automation to support data analysis modeling process.

1.5 Thesis Outline

In the rest of this thesis, Chapter 2 presents some of the existing related work related to this work, including data analysis modeling process, variability modeling analysis, feature models, and object-oriented frameworks. Chapter 3 describes the variability-aware design approach proposed for CRISP-DM data analysis modeling process, which includes the feature diagrams and a preliminary framework design. Finally, Chapter 4 presents conclusions and outlines future work directions.

Chapter 2
Related Work

This chapter provides details about the research topics related to this thesis, which include data analysis and its modeling process, variability applied to the modeling process, and object-oriented frameworks. We start by providing an overview of the existing data analysis process approaches and present details of the CRISP-DM model process. Then, we describe current research approaches related to variability modeling. Finally, we discuss the topic of object-oriented frameworks.

In general, our approach differs from existing research proposed in the literature in that:

(i) No assessment of the variabilities associated with the CRISP-DM data analysis modeling process has been provided.

(ii) There is a need for software framework designs based on a more comprehensive variability assessment.

(iii) Opportunities for automation of the CRISP-DM data analysis modeling process have not been explored based on a variability-aware perspective.

2.1 Data Analysis Processes

The main goal of data analysis is to acquire knowledge from data for several purposes such as understanding data, decision making or problem solving. The data analysis process essentially comprises preparation, preprocessing, analysis, and post-processing, and throughout all of its phases this process relies on human expertise with different skills and knowledge in areas that include statistics, computer science, and information systems. Data volume has increased exponentially over the years, making automated tool support inevitable.

Several data analysis processes have been proposed by the industry and academia to describe the steps that data analysis experts should follow when solving their problems, including CRISP-DM developed by a consortium of companies (i.e., NCR, SPSS, and

Daimler-Benz) and SEMMA developed by the SAS Institute [9], [10], [11], [12]. Specifically, CRISP-DM has modeling as one of its phases, which involves selecting a modeling technique, generating a test design, building a model, and assessing the model.

2.1.1 Existing Data Analysis Processes

A number of process models and methodologies have been proposed to support data analysis process. Mariscal et al. presented a comparative study review of the 14 most used process models and methodologies for data analysis (e.g., data mining and knowledge discovery) [43].

A pioneering approach for the data analysis process was the KDD process model, which was the foundation for many other approaches, including CRISP-DM, considered the *de facto standard* model [9].

Figure **2** illustrates the evolution of the data analysis methodologies [43], which shows KDD as the start point and the analysis process models derived from it.

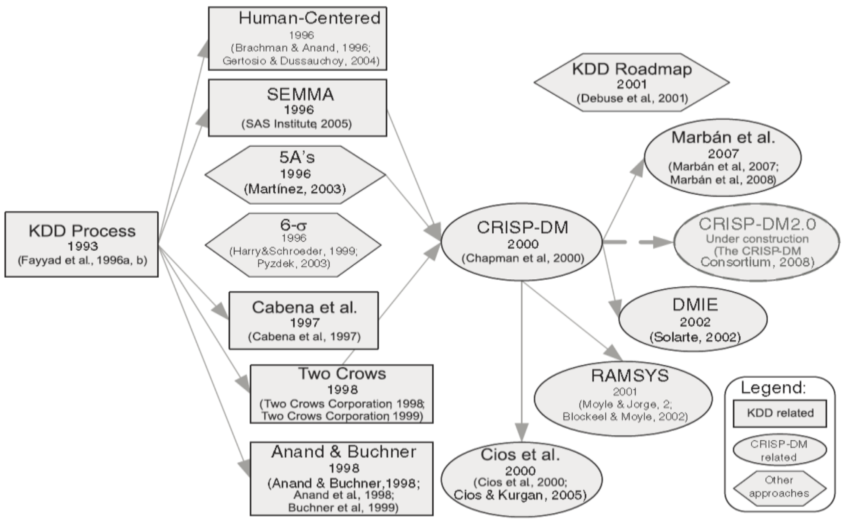

Human-Centered
1996
(Brachman & Anand, 1996; Gertosio & Dussauchoy, 2004)
SEMMA
1996
(SAS Institute, 2005)
5A's
1996
(Martínez, 2003)
6-σ
1996
(Harry&Schroeder, 1999; Pyzdek, 2003)
KDD Process
1993
(Fayyad et al., 1996a, b)
Cabena et al.
1997
(Cabena et al, 1997)
Two Crows
1998
(Two Crows Corporation 1998; Two Crows Corporation 1999)
Anand & Buchner
1998
(Anand & Buchner,1998; Anand et al, 1998; Buchner et al, 1999)
KDD Roadmap
2001
(Debuse et al, 2001)
Marbán et al.
2007
(Marbán et al, 2007; Marbán et al, 2008)
CRISP-DM
2000
(Chapman et al, 2000)
CRISP-DM2.0
Under construction
(The CRISP-DM Consortium, 2008)
DMIE
2002
(Solarte, 2002)
RAMSYS
2001
(Moyle & Jorge, 2; Blockeel & Moyle, 2002)
Cios et al.
2000
(Cios et al, 2000; Cios & Kurgan, 2005)
Legend:
KDD related
CRISP-DM related
Other approaches

Figure 2 - Evolution and relationship among data analysis process models (from [43]).

As a process model, CRISP-DM provides an overview of the data analysis life cycle and its phases. In Table 1, several data analysis process approaches are compared with CRISP-DM, in terms of the phases each approach can support [43].

	APPROACHES					
	Human Centered (1996)	**SEMMA (1996)**	**Cabena et al. (1997)**	**Two Rows (1998)**	**Anan & Buchner (19998)**	**CRISP-DM (2000)**
PHASES	Text discovery	-	Select	Define business problem	Domain knowledge elicitation	Business understanding
	Data discovery				Human resource identification	
					Problem specification	
	Text discovery	Sample Explore	Preprocess	Build DM Explore Data	Domain knowledge elicitation	Data understanding
	Data discovery				Data prospecting	
	Data cleaning					
	Data cleaning	Explore Modify	Transform	Explore data for modeling Prepare data	Methodology identification	Data preparation
	Model development				Data pre-processing	
	Model development data analysis	Model Assess	Mine	Build model	Methodology identification Pattern discovery	MODELING
	Data analysis	Assess	Analyze Assimilate	Evaluate model	Knowledge post-processing	Evaluation
	Output generation	-	Analyze Assimilate	Deploy model and results	-	Deployment

Table 1 - CRISP-DM vs. other data analysis process approaches.

2.1.2 CRISP-DM

CRISP-DM is a standard data analysis process model and an associated methodology created in 2000 by a consortium of companies, (i.e., NCR, SPSS, and Daimler-Benz) [9]. The main goal of this process model was to consolidate approaches and ideas related to the data analysis process through a standard, organized, and structured approach. It was designed to be "industry-, tool-, and application-neutral" [9].

This model proposes a pre-defined sequence of phases but there may be a need to repeat some of the phases or some tasks defined within each phase in specific situations. The overview lifecycle is meant to help beginners in data mining to understand the whole data analysis process and what to do to in each phase of this process.

As shown in Figure 3, the data analysis process model proposed in CRISP-DM is based on a cyclical approach that involves six phases [9], [11]. The arrows indicate the most common and meaningful sequences of phases that are typically applied in the data analysis process, however moving back and forth between phases is expected in some cases [9], [11].

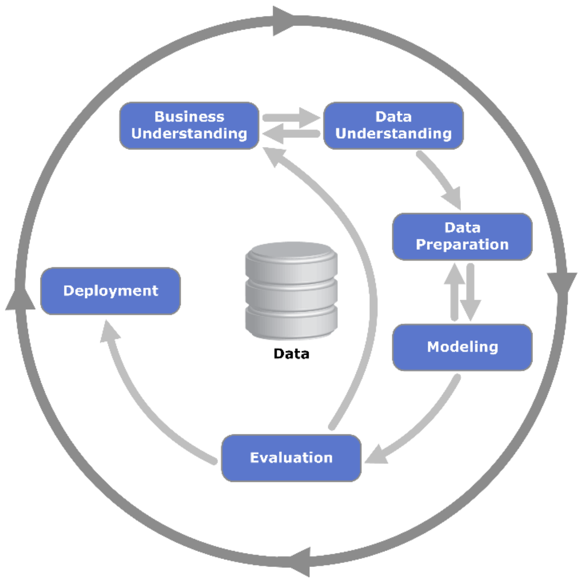


Figure 3 - CRISP-DM data analysis phases (from [9]).

According to the CRISP-DM Reference Guide [9], the process model is hierarchically organized into four levels, namely phases, generic tasks, specialized tasks, and process instances. The higher level of the data analysis process comprises six phases that are executed not necessarily in a pre-defined sequence [9]. The generic tasks refer to activities that can be applied to general data analysis applications. The specialized task level describes how to perform the actions defined in the generic level. The pre-defined sequence for executing the phases and tasks is flexible and constitutes a typical execution order. In a specific situation, this sequence can be changed and it may be necessary to repeat tasks defined in previous phases. The process instance level consists of a record of the decisions, actions and results of the data analysis applications.

The six phases of CRISP-DM are:

- Business Understanding

The "Business Understanding" phase involves the understanding of requirements, needs, and objectives of the project from a business perspective, in order to define a data mining problem and design a preliminary plan to achieve these objectives.

- Data Understanding

The "Data Understanding" phase involves the collection of initial data. The activities of this phase involve getting familiar with the data, identifying problems in data quality, discovering the first insights from data, and generating hypotheses.

- Data Preparation

The "Data Preparation" phase involves the activities needed to build the final dataset to be used by the modeling tool(s). The tasks in this phase can be executed multiple times, and not necessarily in the typical sequential order. This phase usually takes 50% to 70% of the entire time and effort spent in a data analysis process.

- Modeling

The "Modeling" phase involves the selection and application of several modeling techniques. Because many techniques can be applied to the same data analysis problem, this phase requires performing several activities, such as parameter calibration to obtain optimal values, test design definition, model building, and model assessment. Some of these activities may require previous tasks, including those in the data preparation phase or the business understanding phase, to be performed again.

- Evaluation

The “Evaluation” phase involves the evaluation whether the data analysis model results achieve the business objectives, and the quality and success criteria. This is the key task for decision-making based on the data mining results.

- Deployment

The “Deployment” phase involves the organization and presentation of the data analysis results in a way that end users can understand and use for decision-making purposes. This task constitutes the closing phase of the data analysis process. Depending on the data analysis requirements, this phase may produce a simple report or a more complex company-wide implementation project.

2.1.3 CRISP-DM Modeling Phase

The data analysis modeling phase is a multi-iteration process, in which several modeling techniques are applied to the same problem and their parameters are fine-tuned to optimum values until specific quality criteria are satisfied. Depending on the model chosen, it might be necessary to return to the previous phase, data preparation, for other rounds of transformation and cleaning that are required for that model [11].

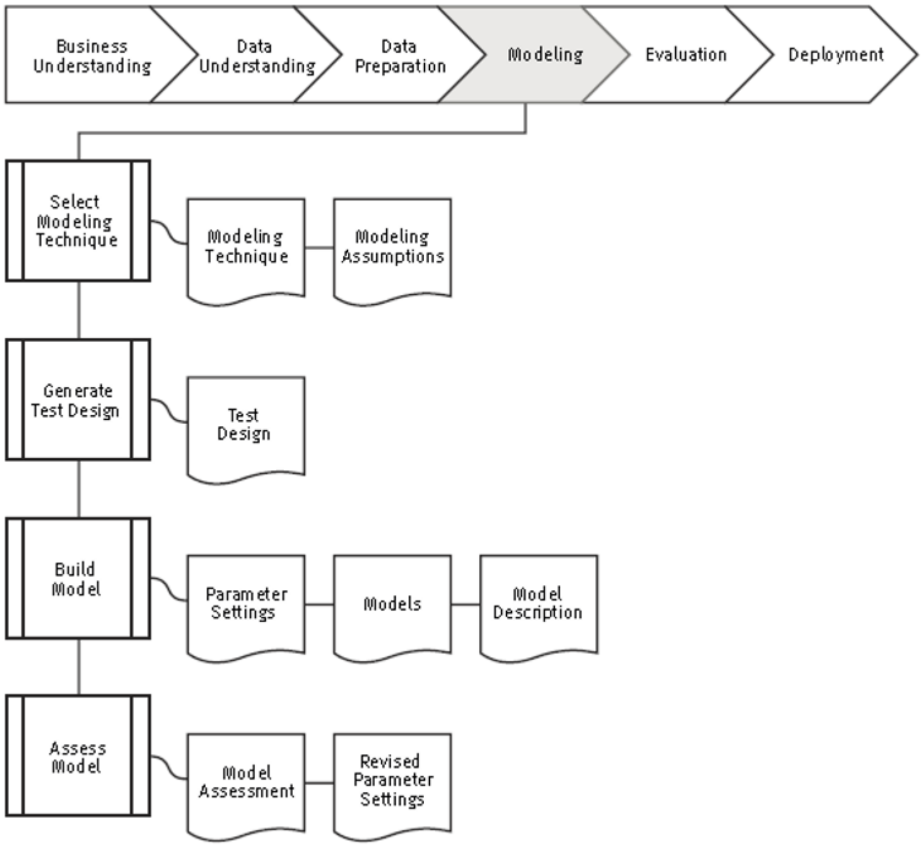


Figure 4 - CRISP-DM data analysis modeling generic tasks (from [9]).

According to the CRISP-DM process [9], [11], the modeling phase is where models are created through the application of modeling techniques and algorithms to extract interesting data patterns, correlations, and associations. For this purpose, this phase consists of four generic tasks: (a) Select modeling technique; (b) Generate test design; (c) Build model; and (d) Assess model, as illustrated in Figure 4.

- Select Modeling Technique

This task refers to the selection of a specific modeling technique. This should be performed separately for every technique applied, and comprises two outputs: the modeling techniques and the modeling assumptions.

- Generate Test Design

This task refers to the generation of a procedure to test the model quality and validity needs, before building a model. In general, a dataset is divided into training and test sets. The model is built based on the training set, and its quality is estimated based on the test set. This task has only one output, the test design, which includes the description of the plan for training, testing and evaluating the models.

- Build Model

This task refers to the execution of the modeling tool using as input the prepared dataset to create the models. The outputs are the models produced by the selected tool, the parameter settings, and the model description.

- Assess Model

This task refers to the assessment of the models based on the domain knowledge, the data analysis success criteria, the test design, and the decision on which models are both accurate enough and effective to be used in the next phases. The outputs of this task are: model assessment and the revised parameters settings. The success of the application of modeling and discovery techniques are evaluated and the data analysis modeling results are presented to domain experts and business analysts.

2.1.4 CRISP-DM Extension Approaches

Since CRISP-DM was launched in 2000, it is one of the most widely adopted process models for data analysis processes [9], [44] and the most complete [45]. However, some aspects of the data analysis process are not supported by CRISP-DM, including the ability to deal with temporal and contextual data. For this reason, some of the approaches to extend CRISP-DM were proposed in the literature, including CRISP-TDM [46], CAPS-DM [47], CRISP-MED-DM [48], and CRISP-EM [49].

The study conducted by C. Catley et al. [46] revealed that the CRISP-DM methodology presents limitations when applied to temporal data such as real-time, high-frequency, multidimensional time series data, and proposes an extension, named CRISP-TDM, to adapt specifically the original phases 1, 2, 4 and 6, so that modeling systems based on Intelligent Data Analysis (IDA) could appropriately support temporal data mining (TDM). CASP-DM [47] is another approach that extends CRISP-DM, specifically to make the methodology context-aware, since context abstractions allow the activities in all phases of the data analysis process to be customized to specific situations. O. Niaksu [48] proposes an extension for medical domain called CRISP-MED-DM that addresses typical challenges in medical data analysis, such as the need to deal with inaccurate and fragmented information. In [49], J. Venter, A. de Waal and C. Willers present CRISP-EM (Cross-Industry Standard Process for Evidence Mining), a specialized CRISP-DM approach based on evidence mining to support digital forensic investigation.

2.2 Variability Modeling and Analysis

Variability applied to software engineering enables the adaptation of the structure, behavior, and development process of a software system, and affects its quality and functionality [17], [50]. Variability helps to support the gradual evolution of a system software component and enables innovation opportunities in system development [50]. Variability models can be represented by different models such as the feature model [24] and the UML based model [27].

2.2.1 Feature Models

Commonalities and variabilities of a product can be captured in abstract ways using entities called features [52]. The notion of features is described through several definitions. To cite some, according to K. Kang et al., a feature is “a prominent or distinctive and user-visible aspect, quality, or characteristic of a software system or systems” [53]. K. Czarnecki et al. extend this defining feature as “a property concept, which is relevant to some domain stakeholder and is used to discriminate between concept instances” [54]. Overall, features are abstractions significant to stakeholders that can be combined to express variabilities and commonalities of a product and defined using feature models [55].

Feature models were first introduced by K. Kang et al. in the FODA method to express software product commonality and variability, and the dependencies between them [55]. Since then, they have been widely applied to Software Product Lines (SPLs) and several methods (e.g., FORM, which has been proposed to extend some original concepts, and is able to represent concepts such as feature and group cardinalities, attributes and diagram references [56].

A feature model provides an abstract view of the variabilities and commonalities of requirements in a domain, enabling the definition of features, their properties, and their relations [56]. They are used to understand, define, and communicate the variability of a software system. In addition, feature models help to support the development of common architecture and components, and constitute a key approach to plan for reusability [57]. Feature models provides two types of relationship between features: one defines the relation between a parent feature and its child features and are represented by feature diagrams, and the other, called cross-tree relationship, defines the “requires” and “excludes” relationship in the model.

Feature models are graphically represented in a hierarchical diagrammatic notation, called a feature diagram, which expresses logical relationships between the features [55], [53]. As presented in Figure 5, a feature diagram is depicted as a tree whose starting point is the root feature, which is also called the concept feature or parent and has no concrete implementation. The connections between a parent feature and the group of child features are categorized as dependency relations that express possible composition rules, as presented in Table 2 [58]. The semantics applied to feature models specify that once a feature is selected, its parent is selected too. If a parent is selected, all mandatory features that are in AND-Group are selected as well. If a feature is in an OR-Group, at least one feature must be selected. If it is in an ALTERNATIVE-Group, exactly one feature is selected [53].

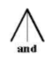
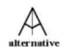
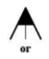
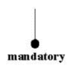
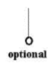
Notation	**Description**	**Symbol**
AND	All features must be selected	
ALTERNATIVE	Only one feature can be selected	
OR	One or more can be selected	
MANDATORY	Features that are common to all instances	
OPTIONAL	Features that are optional	

Table 2 - Feature diagram notation.

In the example Figure 5, the Concept feature consists of two main features, one optional and other mandatory. The Mandatory Feature indicates that the feature is required and common to all instances of the domain, while the Optional Feature is not necessarily part of all instances. The Optional Feature consists of two Or-Group features, indicating that at least one of the two OR Features must be selected. The Mandatory Feature consists of two Alternative-Group features, representing that exactly one of the Alternative Features has to be selected.

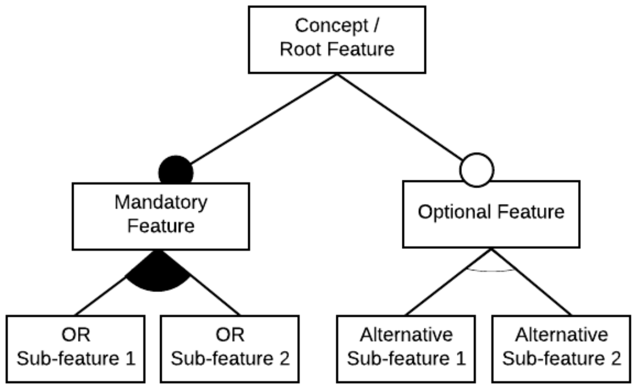


Figure 5 - Example of a feature diagram.

Another example is depicted in Figure 6, in which car is the concept feature. It consists of three main features: transmission, horsepower, and air conditioning. The features transmission and horsepower are mandatory features, showing that they are required and common to all instances of the domain. The feature air conditioning is optional because it is not necessarily part of all instances. The feature transmission consists of two alternative features, manual and automatic, because a car can either have a manual or automatic transmission system.

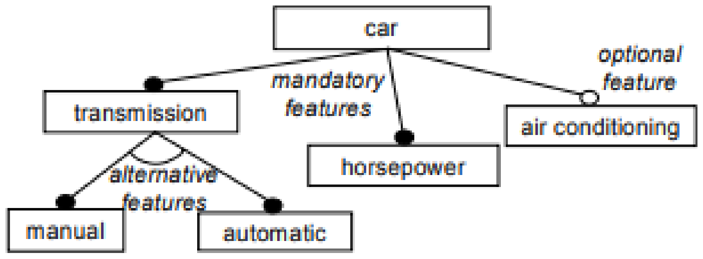


Figure 6 - Example of a feature diagram from [53].

Not all relations between features are represented in the feature diagram. Relations can often be complemented and ambiguities removed by assigning cross-tree constraints and grouping them to the features [59], [60]. Cross-tree constraints are defined by constraints among features and are not expressed hierarchically [55]. They are propositional formulas that must be true. There are mainly two kinds of cross-tree constraints, which are called requires and excludes constraints. They are defined so that, given two features X and Y:

X requires Y, defined as the proposition ($X \rightarrow Y$), where $\rightarrow$ is logical implication: if X is included then Y must also be included (X excludes Y, defined as the proposition $\neg (X \land Y)$, $\neg$ is negation and $\land$ is conjunction: if X is included, then Y is not allowed to be included, and vice-versa.

Using these two kinds of constraints, cross-tree constraints in a specific feature diagram can be represented, although in the general case arbitrary propositional formulas can be used [58].

2.2.2 UML Modeling

In the case where variabilities are modeled using UML, specifically UML class diagrams, the class diagrams and the variabilities can be derived manually using nouns and verbs [42]. In contrast, some techniques have been proposed for the automated generation of UML class diagrams from documents in natural language. Overall, these techniques still rely heavily on user input. D. Kumar proposed a domain independent approach and supporting tool, named, UML Model Generator from Analysis of Requirements (UMGAR), which generates UML models such as class diagrams, use-case diagrams and collaboration diagram using Natural Language Processing (NLP) techniques [61]. L. Mich developed a system called LOLITA to generate an object model automatically from natural language. The approach assumes that nouns are objects and uses links to find relationships among the identified objects. The approach, however, can identify objects but not classes, and cannot identify classes or distinguish between classes, attributes and objects [51]. In addition, Song et al. introduced a taxonomic class modeling methodology (TCM) to support object-oriented analysis based on rules, such as rules for noun analysis, English sentence structure, class categories and checklists. According to their methodology, candidate classes are initially identified based on noun analysis, and then some of these classes are eliminated using elimination rules and other classes are discovered using pre-defined class categories [62].

Several other approaches to modeling and analyzing variability have been proposed in the literature [63], [64], [50]. MSinnema et al. propose a framework for variability modeling called COVAMOF, whose view uniformly represent features, architecture, and component implementation [63]. This approach represents all variation points and dependencies as first-class entities. J. Horcas et al. applied variability modeling to help developers perform an analysis of different options of functional quality attributes, in terms of energy efficiency and performance for a given application [65]. In [66], a framework based on a variability model was used to guide the dynamic adaptation of service compositions at design time and at runtime. Furthermore, C. Ayora et al. proposed a framework to evaluate process variability and to compare different process-based variability approaches [68]. Another variability approach is proposed by M. Koning et al., in which variability was incorporated to provide flexibility in a Web service system [69].

2.3 Object-Oriented Frameworks

An object-oriented framework can be defined as “a set of classes that embodies an abstract design for solutions to a family of related problems” [31]. Several frameworks have been proposed in the literature to support the data analysis process. Some of these frameworks are TMiner, IRIS, and JStatCom.

F.Berzal et al. propose the TMiner component-based data analysis framework, which was used in several application domains, to support the entire KDD process in a single general-purpose framework [70]. Another framework, IRIS, is a modeling framework for business process reengineering based on big data analytics and a goal-oriented approach. The approach comprises a conceptual modeling language, a process representation and a tool for effective business processes reengineering. In addition, S. Wang and C. Eick present a data analysis framework for environmental and geo-spatial data analysis which integrates pre-processing techniques, two density-based clustering algorithms, a post-processing analysis technique, a change pattern discovery algorithm, and visualization techniques [71]. In [72], M. Khanbabaei et al. present an integrated framework for process improvement through data analysis techniques and ontology concepts. Another data analysis framework is proposed in [73], whose goal is to identify intrinsic linear relationships among human factor events in nuclear power plant operations.

Root is an example of a basic object-oriented data analysis framework for large-scale data analysis proposed by R. Brun and F. Rademakers [74]. It offers a common set of tools and features for data analysis through a graphical interface, a command line or batch scripts. B. Ulfenborg et al. present a framework for biomedical big data analytics, which is applied to the analysis of transcriptomic time series data from early differentiation of human pluripotent stem cells towards the mesoderm and cardiac lineages [75]. M. Krätzig proposes a software framework for data analysis called JStatCom [76]. This framework defines classes specially designed to link existing math libraries and a Java client, providing convenient user interface components.

Chapter 3
A Variability-Aware Data Analysis Modeling Process

This chapter presents the core results of this thesis, which includes the description of a variability-aware data analysis modeling process in terms of:

(i) The assessment of the variabilities inherent in the CRISP-DM data analysis modeling phase and the provision of feature models that represent these variabilities.

(ii) The definition of a preliminary framework design that captures the identified variabilities.

(iii) The evaluation of possibilities for automating the data analysis modeling process.

3.1 Variability Assessment

The scope of the variability assessment is the modeling phase of the CRISP-DM model process, defined according to the CRISP-DM Reference Guide [9]. The variability assessment is conducted for each of the four generic tasks of the modeling process, namely Select Modeling Technique, Generate Test Design, Build Model and Assess Model.

In this section, as illustrated in Figure 7, the identification of variabilities and the features related to the CRISP-DM data analysis modeling phase are provided and these variabilities are represented as feature diagrams. These diagrams describe the logical relationship between the features present in this modeling phase. Although, logical relationships are captured in the diagrams, cross-tree relationships are not represented as a complement to these diagrams. The inclusion of cross-tree relationships, which include assertions about features requiring or excluding other features, require domain-specific knowledge about a wide variety of specific techniques and algorithms and about the feature interactions of these methods and algorithms adopted in technique selection with those that can be used in testing, model building, and model assessment.

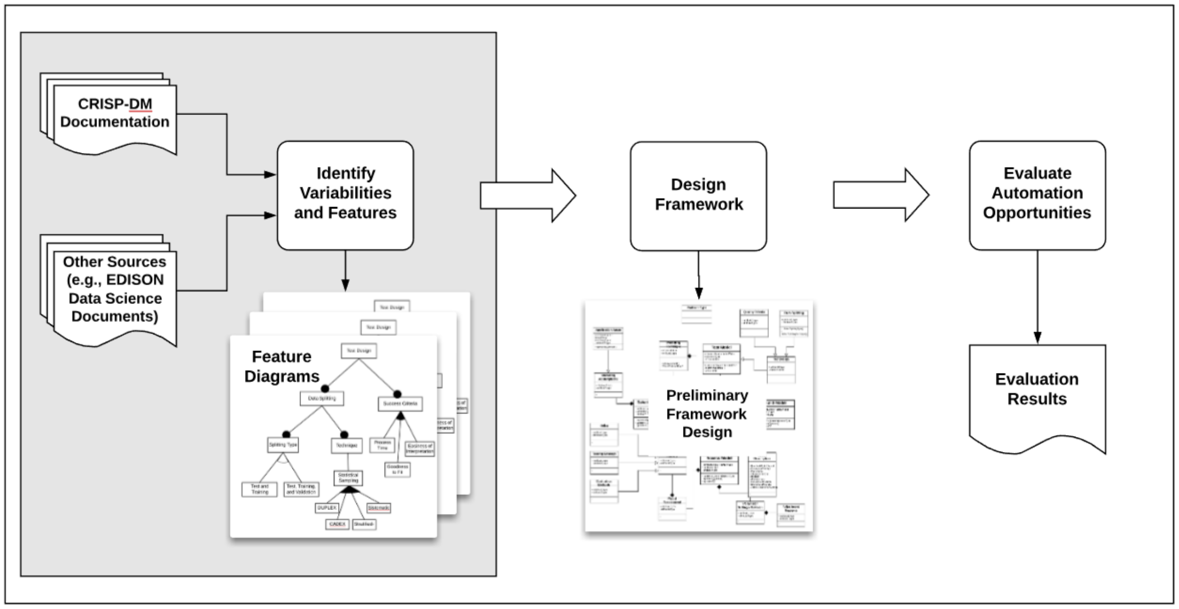

CRISP-DM Documentation
Other Sources (e.g., EDISON Data Science Documents)
Identify Variabilities and Features
Feature Diagrams
Design Framework
Preliminary Framework Design
Evaluate Automation Opportunities
Evaluation Results

Figure 7 - A variability-aware design approach: Variability assessment.

The assessment of variability starts with the analysis of the CRISP-DM documentation [9], [11] to understand the CRISP-DM data analysis modeling process. Next, we thoroughly investigated all the steps, generic tasks, specialized tasks, and outputs of the modeling phase to identify nouns, actions, verbs, and results of every task. We proceed with an informal evaluation of potential variations for each one of these items. An item might represent a variation point whenever it characterizes design options. For example, when we find “select method,” because “method” is a noun, this can indicate that method can be a variation point that has as design options the several methods that can be adopted in the design. In addition, following the same example, because “select” is a verb, this may indicate that “select” or “selection” can be a variation point that has as design options the several possible ways through which the method can be selected. A preliminary list of the main categories for the CRISP-DM data modeling phase is presented in Appendix A. A variation point can lead to other variation points or ultimately to a final variant in the variation point hierarchy.

Part of the variability analysis, specifically the one related to the “Select Modeling Techniques” task, was mainly based on the EDISON Data Science documents [77], [78].

This data science documentation was developed in a research project called the EDISON Project. This project was created to establish a standardization in data related competencies, such as professional skills, technological concepts, models, and knowledge areas. The framework is intended to provide a basis for the data science supply-demand-community ecosystem and includes the following components [77]:

(i) Data Science Competence Document (CF-DS).
(ii) Data Science Body of Knowledge (DS-BoK).
(iii) Data Science Model Curriculum (MC-DS).
(iv) Data Science Professional Profiles (DSP).

Our research takes into consideration the knowledge area classification for data analysis proposed in EDISON to support the assessment of variabilities related to approaches, methods, models, and algorithms, in the task “Select Modeling Technique.” Additional classifications provided in the literature contributed to enrich the variability analysis for the other tasks [79], [80], [81], [82], [83], [84].

This section addresses the first research question: What are the variabilities related to the CRISP-DM data analysis modeling process and how can these variabilities be represented using feature models?

The CRISP-DM Reference model [9] describes modeling as the phase in which “various modeling techniques are selected and applied, and their parameters are calibrated to optimal values.” The feature diagram presented in Figure 8 represents the hierarchical structure of the refined properties for the modeling phase, whose concept feature is so called *Modeling*.

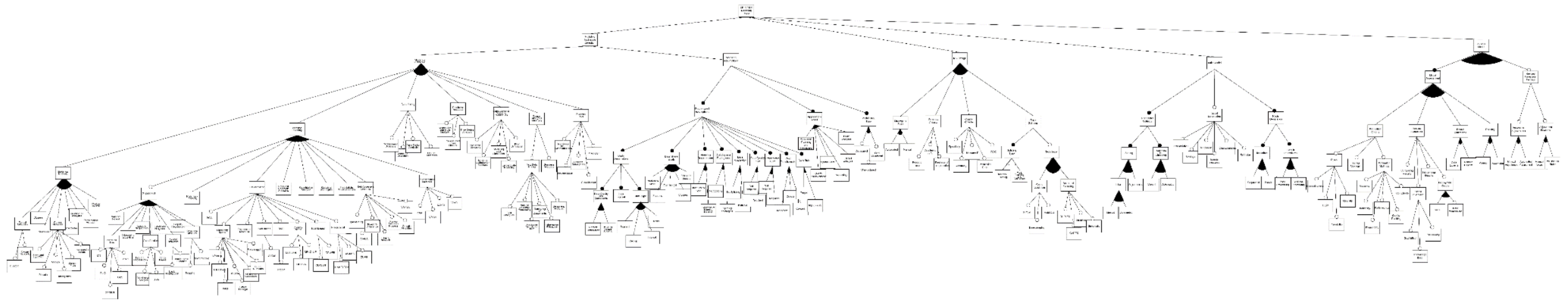


Figure 8 - Feature diagram for the CRISP-DM modeling phase.

The modeling process consists of four generic tasks, namely "Select Modeling Technique," "Generate Test Design," "Build Model" and "Assess Model" that are common characteristics of every modeling phase. Therefore, the four main features of the *Modeling* concept feature are defined as the following features: *Technique Selection, Test Design, Build Model,* and *Assess Model*. These four features are mandatory with respect to the parent concept (*Modeling*) because they must be performed for every modeling phase process.

In the following paragraphs, explanations are provided on how each of the four features are decomposed according to the CRISP-DM documentation [9], [11]. Each decomposition follows the feature diagram creation process defined previously in Section 2.2.1.

The first task is the "Select Modeling Technique." This step describes the process of defining the most appropriate model types to use to address a data mining project needs. The corresponding feature that express this instance is named *Modeling Technique Selection*. Figure 9 shows the feature diagram related to *Modeling Technique Selection*. The outputs produced in this step are "Modeling Technique" and "Modeling Assumptions" and to express them in the feature diagram, two features named *Modeling Technique* and *Assumptions* are included. The decomposition for these features considers the activities described for this task in [9], which involve: (a) the decision on which technique can be appropriately used, bearing in mind the tool selected; (b) the definition of any built-in assumptions made by the technique about the data (e.g., quality, format, distribution); (c) the comparison and validation of the assumptions defined in the previous phases. These instances are required to occur, then both features *Techniques* and *Assumptions* are related through a mandatory relationship with the feature *Modeling Technique*.

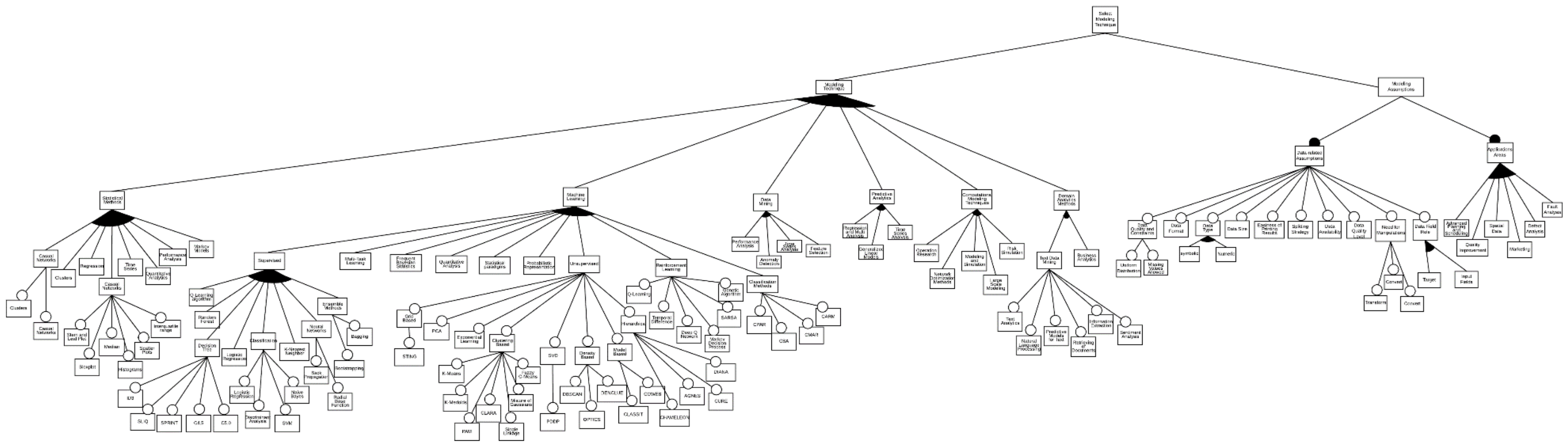


Figure 9 - Feature diagram for feature Select Modeling Technique.

The feature *Selecting Modeling Techniques* is included in the diagram as an or-group feature. It expresses the process of selecting a modeling technique, which mainly involves deciding "an appropriate technique for the exercise, bearing in mind the tool selected" [9]. In addition, according to [11], when deciding on which model(s) to use, some issues should be considered, such as: (a) data splitting technique; (b) data availability to produce reliable results for a given model; (c) quality level of current data; (d) data type appropriateness for a particular model; and (e) data conversion needs. Thus, choosing a technique to generate a model takes into consideration elements that may be combined into several multiple categories such as data analysis modeling approach, modeling technique types, model types, and algorithm types. The EDISON project [77] provides data analysis (the so-called data science) subject domain classifications based on existing standard, commonly accepted approach categorizations, and other publications from industrial and research communities. In this context, it proposes created data analysis knowledge areas and related sub-classifications. Based on that structure, the first level of or-group feature corresponds to the data analysis modeling approaches namely *Statistical Methods, Machine Learning, Data Mining, Predictive Analytics*, *Computational Modeling Techniques*, and *Domain Analytics Methods*. The second level is also based on the EDISON project [77] and refers to modeling technique types. For example, the feature *Machine Learning* is a parent feature whose child features depict the group of modeling technique types which include types such as Supervised, Unsupervised, and Reinforced Learning. The third level of child features represents the model types. For example, for the feature *Supervised*, some of features include *Decision Tree, Naive Bayes, Ordinary Least Square Regression, Logistic Regression, Neural Networks, SVM, Ensemble Methods*, and others. The last child feature refers to the algorithm type, and some examples represented in the feature diagram are the *Apriori Algorithm, ID3, C5.0,* and *SVM*.

The mandatory feature *Modeling Assumptions* consists of a group of alternative features which specify assumptions about data according to the modeling technique selected [9]. The activities for this step involve [9]: (a) defining any built-in assumptions made by the technique about the data (e.g., quality, format, and distribution); (b) comparing these assumptions with those defined previous phases; (c) ensuring that the assumptions hold and go back to the previous phase, if necessary. Determining an appropriate model would also consider the availability of data types for mining, the data mining goals, and the specific modeling requirements [11]. These elements are optional and might occur depending on the technique selected. Based on this, the feature diagram depicts or-group features, namely: *Data-related Assumptions, Application areas,* and *Model Data Type*.

The mandatory group feature *Data-related Assumptions* expresses elements that support decision-making process regarding data type, depending on the selected model [11]. In the feature diagram it is depicted by a structure that includes the features: *Data Format, Data Quality and Constraints, Data Type, Specific Modeling Requirements, Splitting Strategy, Data Availability, Data Quality Level, Need for Manipulations,* and *Data Field Role*.

The feature *Data Quality and Constraints* generates two optional features, namely *Uniform Distributions* and *Missing Values Allowed*. According to [11], specific modeling requirements should be considered when determining the model type to use. Some requirements cited in [11] refer to data size or type required by a model and the need to have results easily presented. The feature *Data Type* refers to the types of data to be used when applying a specific modeling technique and it is represented as an or-group of features which include [11]: *Symbolic* and *Numeric*.

Another aspect that has to be considered is the way the data can be split which is expressed in the diagram as an optional feature named *Splitting Strategy*. The need for data to "produce reliable results for a given model" [11] is another variability in the data-related assumptions. It is referred to as data availability and is represented as an optional feature named *Data Availability*. According to [9], data quality is another variation point that is related to the technique and the data quality level for that technique. This aspect is represented by the feature named *Data Quality Level*.

The need for data manipulations or transformations is another issue that might impact the model choice [11], [9]. To meet the model requirements data might need to be transformed, converted, or rebalanced. For that situation, this variation point and its variants are represented in the diagram as feature *Need for Manipulations* and have a relationship with a set of optional features that indicate a need for preparing the data before execution namely *Transform, Convert,* and *Rebalance*.

The feature *Application Areas* consists of an alternative or-group which represent the application domain that the data mining project refers to. The variability for this element is expressed in the feature diagram as a set of optional alternative feature examples named *Advanced Planning and Scheduling (APS), Quality Improvement, Spatial Data, Marketing, Defect Analysis,* and *Fault Diagnosis*.

To generate the feature diagram for "Test Design," the description of the "Generate Test Design" task was considered. This task describes the procedure of testing the model results in terms of quality and validity. This procedure is applied to as many models as needed and depends on the number of models selected to be used and deployed. The output of this step is a comprehensive test design that is "the intended plan for training, testing, and evaluating the models" [9]. This test design describes "the criteria for goodness of a model" and defines "the data on which these criteria will be tested" [11]. As stated in [9], the activities for this step involve checking the existing and appropriate test designs for each data mining goal, preparing data required for the test, and deciding on necessary steps, such as number of iterations or number of folds.

Based on the previous description of the "Generate Test Design" task, the feature *Test Design* consists of three mandatory features: *Strategy to Stop, Test Criteria, and Data Splitting*. Figure 10 shows the diagram related to the feature *Test Design*.

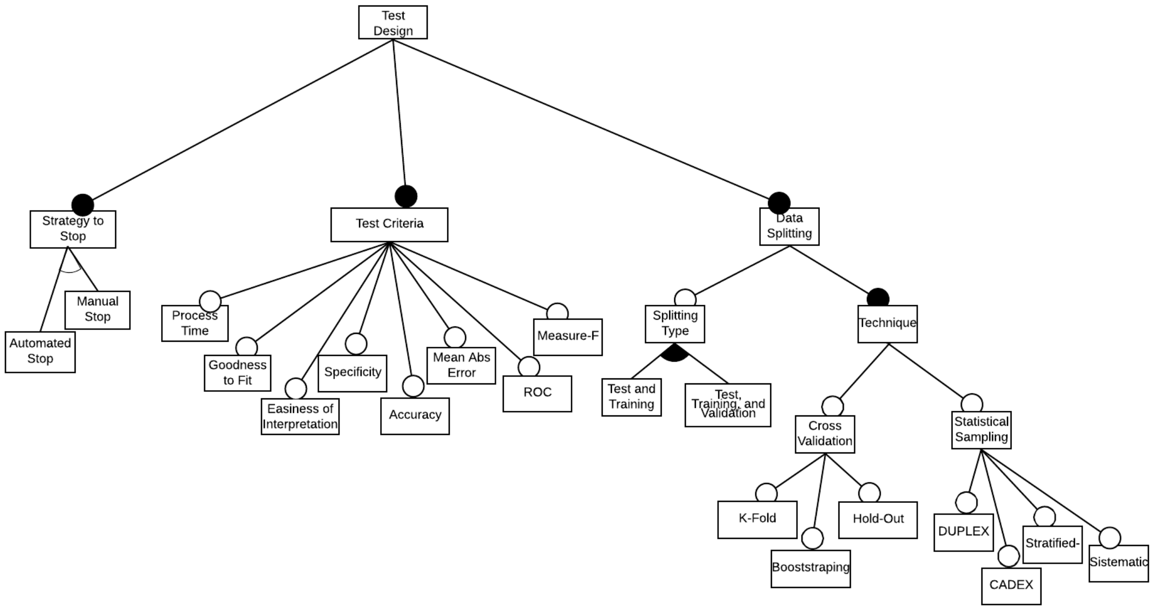


Figure 10 - Feature diagram for feature Test Design.

The feature *Test Criteria* was defined as a mandatory feature because "it is necessary to define a procedure to test the model's quality," as stated in [9]. The quality of a model is measured based on characteristics defined in the literature [49], [69], [70], such as sensitivity, accuracy, specificity, and Mean Absolute Error. Other criteria may vary according to the modeling technique. For example, "for unsupervised models, such as Kohonen cluster nets, measurements may include criteria such as ease of interpretation, deployment, or required processing time" [11]. These features also form an or-group feature. These examples of metrics form an or-group feature, indicating that one or more can be selected as a metric for model quality measurement and success criteria, including: *Process Time, Goodness to Fit, Easiness of Interpretation, Sensitivity, Accuracy, Specificity*, and *Mean Absolute Error*.

Strategy to Stop is another mandatory feature which is based on the fact that "modeling is an iterative process, it is important to know when to stop adjusting parameters and try another method or model" [11]. For this feature, two alternatives features are derived: *Automated Stop* and *Manual Stop*, of which only one can be selected.

To introduce the Data Splitting feature, it was considered that in the "Generate Test Design" step, a procedure is used to "separate the dataset into train and test sets, build the model on the train set, and estimate its quality on the separate test set" [9]. The need to separate datasets into subsets for different purposes indicates that Data Splitting is a mandatory feature. The primary component of the plan for testing the model involves determining how to partition the available dataset into training, test, and validation datasets [9], [11]. Thus, the data split options form another alternative-group feature, namely *Test and Training,* or *Test, Validation, and Training.*

There is a number of techniques that can be applied to the data splitting process. The *Cross Validation* and *Statistical Sampling* form an or-group feature that describes common approaches to data splitting methods. The feature *Cross Validation* contains child features that represent commonly used types of cross-validation strategies, such as Hold-Out, K-Fold, and Bootstrapping [85]. The feature *Statistical Sampling* represents some of most widely used statistical sampling approaches, including: DUPLEX, CADEX, Stratified Sampling, Simple Random Sampling, Convenience Sampling, and Systematic Sampling [85].

The mandatory feature *Build Model* expresses the variabilities identified in the task "Build Model" as described in the CRISP-DM documentation. Figure 11 shows the diagram related to the feature *Build Model*. This task refers to the creation of one or more models by a modeling tool to support the data-mining decisions by comparing the results and analyzing the notes that were made during the process [9], [11]. It is important to track the progress of a variety of models, keeping "notes of the settings and data used for each model" [11]. By the end of the model-building process, three outputs will be produced: "Parameter Settings," "Model Description," and "Models" [9]. These elements are represented in the diagram as two mandatory features *Parameter Settings* and *Model Generation*, and an optional feature named *Model Description*.

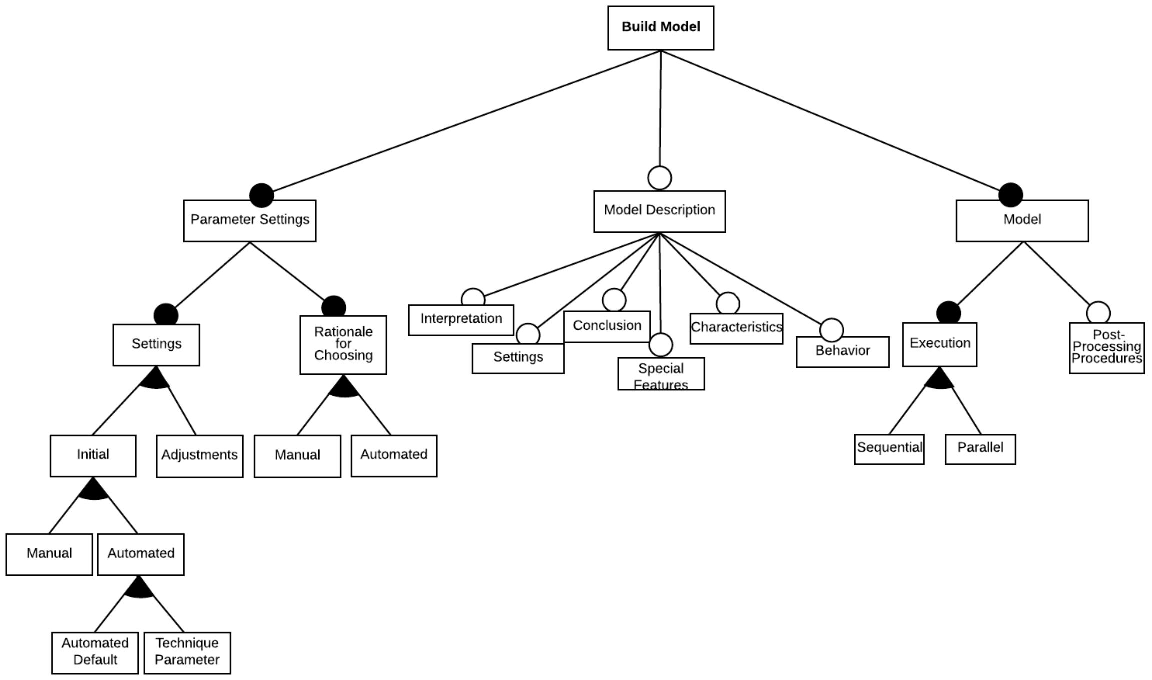


Figure 11 - Feature diagram for feature Build Model.

The output “Parameter Settings” is a list consisting of information about “the initial parameters” and “the reasons for choosing those values” [9], [11]. Therefore, two mandatory features are represented in the diagram to express these two activities involved in the parameter settings [9], which are named *Settings* and *Rationale for Choosing*. According to [9], “there are often a large number of parameters that can be adjusted” and therefore, considering that parameters can be initially set and later adjusted, the feature *Settings* consists of two features related to the moment the setting of parameters might occur. These features are expressed as *Initial* and *Adjustments*. In its turn, the initial parameters can be set automatically or manually, which are represented in the diagram as features *Manual* and *Automated*. To set parameters automatically, it can be preset with default settings parameters or parameters related to the technique applied. Therefore, two exclusive features have relationship with feature *Automated Default Setting* and *Technique Parameter*.

The activity of taking notes of parameter choices and reasons for the choice [9] can be performed manually or automatically. This activity is expressed as the feature *Reasons*, consisting of two exclusive features: *Manual* and *Automated*.

“Model” is another output of the “Build Model” step and refers to the creation of one or more models on a modeling tool. The activities in this task are running the selected technique on the input dataset to produce model and post-processing data mining results (i.e. edit rules, display trees) [9]. The feature *Model* represents the output “Model” and consists of two features named *Execution* and *Post-processing Procedures* to express the activities performed for building the model. The feature *Execution* refers to creation of the models and contains two features *Sequential* and *Parallel*, which express the way the modeling creation may occur. The feature *Post-processing Procedures* represents the post-processing results of the model generation.

The last output is the “Model Description,” which is expressed in the diagram as an optional feature, also named Model Description. According to [9], when examining the results of a model, it is recommended that information be recorded about the modeling experience and its meanings. Activities performed in this task are: (a) describing of characteristics of the current model that may be useful for the future; (b) recording parameter settings used to produce the model; (c) providing a detailed description of the model and any special features; (d) listing the rules produced, plus any assessment of per-rule or overall model accuracy and coverage or list of any technical information about the model (such as neural network topology) and any behavioral descriptions produced by the modeling process (such as accuracy or sensitivity); (e) describing the model’s behavior and interpretation; and (f) providing conclusions regarding patterns in the data (if any); sometimes the model reveals important facts about the data without a separate assessment process (e.g., that the output or conclusion is duplicated in one of the inputs) [9]. In the feature diagram, these activities and properties were expressed as an optional-or feature group consisting of *Interpretation, Parameter Settings, Conclusion, Special Features, Characteristics*, and *Behaviors*.

The last task of the modeling phase is “Assess Model.” The model assessment refers to the analysis of the effectiveness and accuracy of the results to define the final models that will be deployed. The evaluation is based on the criteria generated in the test plan to ensure the model meets success criteria [9], [11]. The activities involved in the model assessment include [9], [11]: (a) evaluating results regarding evaluation criteria; (b) testing result according to a test strategy; (c) comparing the results and interpretation of evaluation; (d) ranking results; (e) selecting best models, (f) interpreting results in business terms; (g) getting comments on models by domain or data experts; (h) checking plausibility of model and effect on data mining goal; (i) checking the model against given knowledge base to see if the discovered information is novel and useful; (j) checking the reliability of the result; (k) analyzing potential for deployment of each result; (l) assessing the rules (in terms of logic, feasibility, quantity, etc) and the results; (m) getting insights into why a certain modeling technique and certain parameter settings lead to good/bad results. Figure 12 shows the diagram that represents the feature *Assess Model*. According to [9], this task generates two outputs: the “Model Assessment” and the “Revised Parameter Settings,” which are represented in the feature diagram as features *Model Assessment* and *Revised Parameter Settings*.

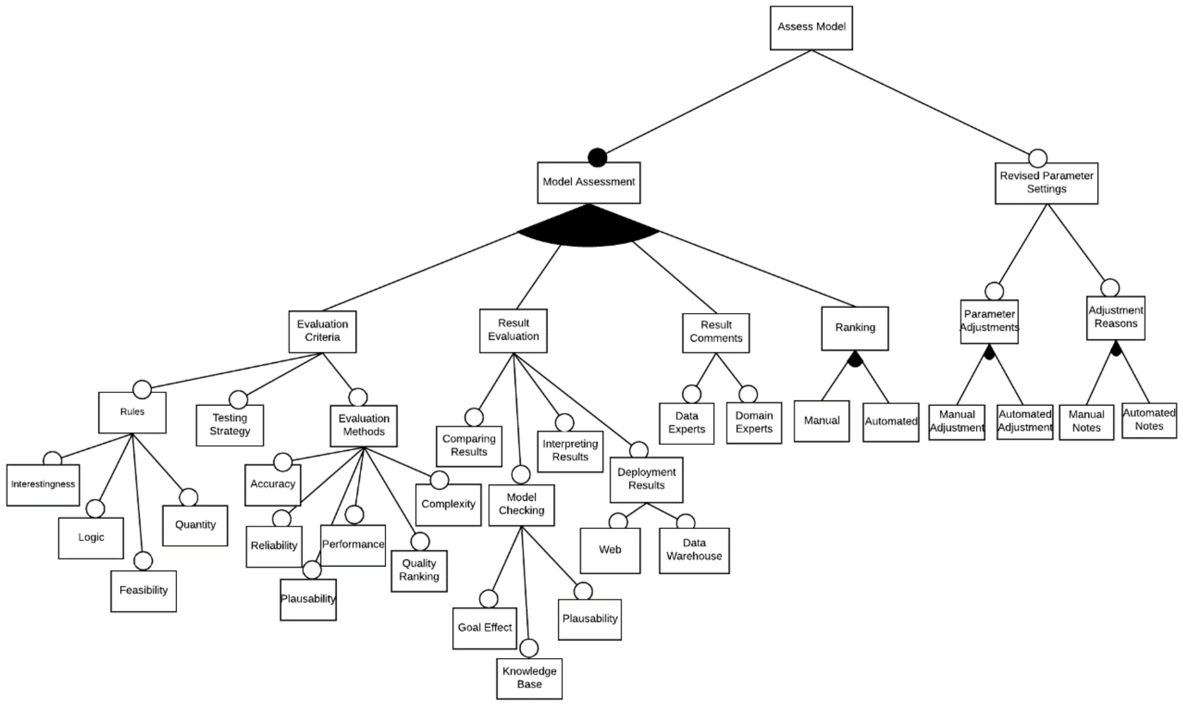


Figure 12 - Feature diagram for feature Assess Model

The "Model Assessment" is a summary of the results, the qualities of the generated model, and the quality rank (in relation to each other) [9]. Taking into account the activity descriptions, the main aspects of the model assessment are expressed in the diagram as three features called *Evaluation Criteria*, *Results Evaluation*, *Result Comments*, and *Ranking*.

The feature *Evaluation Criteria* consists of a group features that communicate the characteristics related to the rules, methods and strategies that can be applied to the evaluation of the model. These elements are represented in the diagram as optional features *Rules*, *Evaluation Methods*, and *Testing Strategy*. According to [11], assessing models require evaluation rules, such as logic, feasibility, quantity, and level of interesting, and methods for checking accuracy, reliability, plausibility, and the quality ranking. So, the feature *Rules* includes a group of optional features consisting of *Logic*, *Feasibility*, *Quantity*, and *Interestingness*. The feature *Evaluation Methods* specifies a group of optional features that represent evaluation criteria methods named *Accuracy*, *Reliability*, *Performance*, *Complexity*, *Plausability*, and *Quality Rank*. The result of the test is evaluated according to the test strategy (Train and Test, Cross-Validation, Bootstrapping, etc) [9]. In the diagram, it is represented as the feature *Testing Strategy*.

The feature *Result Evaluation* expresses the aspects related to the assessment results such as evaluating, testing, comparing, interpreting, and assessing results. These aspects are expressed in a group of optional features that includes *Deployment Results, Comparing Results, Interpreting Results,* and *Model Checking*, which also specifies another group of features named *Goal Effect, Plausability,* and *Knowledge Base*.

The feature *Deployment Results* represents how a model's results are deployed, either over the web or sent back to the data warehouse [11] which specifies two features named *Web* and *Data Warehouse*.

As stated in [9], the model assessment should also consider "comments on models by domain and data experts." This activity is expressed in the feature diagram as an optional feature *Result Comments*, which consists of two features named *Data Experts* and *Domain Experts*.

The ranking of the results can be performed either manually or automatically and, therefore, two alternative features represent the way results can be ranked: *Automated* and *Manual*.

The second output, "Revised Parameter Settings," is generated from an iterative process of adjusting, revising and tuning parameter settings [9]. The main activities refer to revise parameters initially set and record the reasons for adjusting them. These activities are represented in the diagram as two mandatory features, which are called *Parameter Adjustment* and *Adjustment Reasons*. Both activities can be performed manually or in an automated way. Therefore, the ways for performing the adjustment of the parameters are expressed as alternative features named *Manual Adjustment* and *Automated Adjustment*. In its turn, the feature *Adjustment Reasons* can also be executed in two different ways and is represented in the diagram as two alternative feature named *Manual Notes* and *Automated Notes*.

3.2 Framework Design

In this sub-section, as illustrated in Figure 13, based on the feature diagrams presented previously, a preliminary framework design to support the CRISP-DM data analysis modeling phase is defined using a UML class diagram. This framework design is showing in Figure 14.

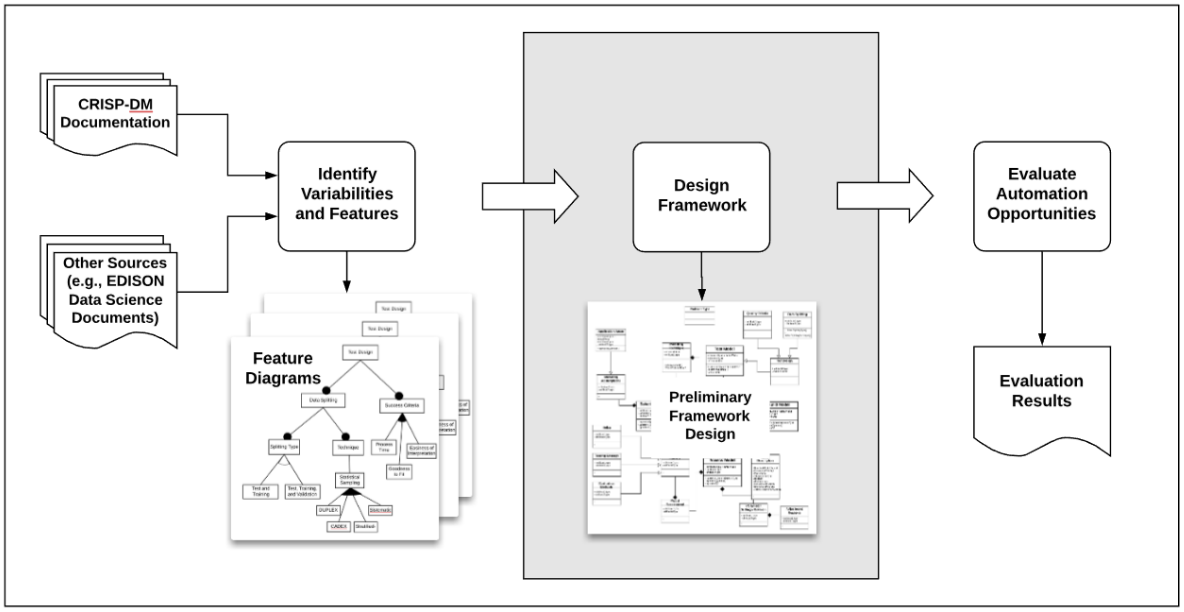

CRISP-DM Documentation
Other Sources (e.g., EDISON Data Science Documents)
Identify Variabilities and Features
Feature Diagrams
Design Framework
Preliminary Framework Design
Evaluate Automation Opportunities
Evaluation Results

Figure 13 - A variability-aware design approach: Framework design.

This framework design represents the variabilities associated with the CRISP-DM data analysis modeling phase for each of its tasks, subtasks and outputs. These variabilities, which were derived informally using nouns and verbs [42], are modeled using UML, specifically UML class diagrams [42]. The framework design is shown in Figure 13. Each variation point is defined as a class in the UML class diagram and the variation points or variants associated with a specific variation point are defined using the class relationship.

Based on the number of leaves of the diagram in Figure 14, the total number of variation options is huge, which indicates that the data analysis modeling process is highly complex and needs to be made easier to perform.

This section addresses the second research question: What preliminary framework design can capture the variabilities in the CRISP-DM data analysis modeling phase?

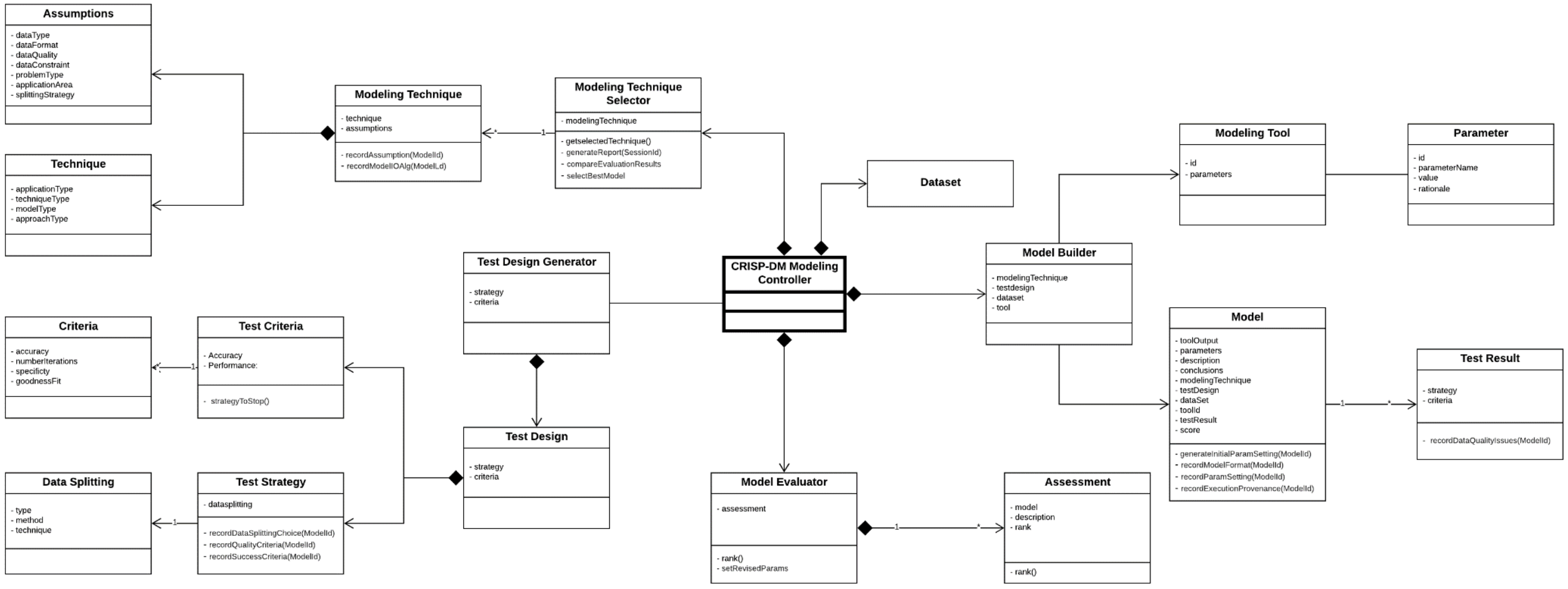


Figure 14 - Framework design for CRISP-DM modeling phase.

The class CRISP-DM Modeling Controller refers to the CRISP-DM Modeling phase and consists of four general sub-systems that represent the generic tasks of the modeling phase. These components are named *Modeling Technique Selector, Test Design Generator, Model Builder,* and *Model Evaluator.*

The *Modeling Technique Selector* represents the first step of the Modeling phase. It provides a class *Model Technique* that expresses the feature *Modeling Technique* as defined in Section 2.2.1. This *Model Technique* is composed of the classes *Technique* and *Assumption* which are the representation of the variation points identified previously. The class *Technique* expresses the variation points and variants related to a technique such as the approach, technique type, model type, and algorithm. To exemplify an instance of Technique, Machine learning can be classified as an approach, Supervised Machine Learning as a type of technique, Decision Tree as a type of a model, and C5.0, a type of an algorithm. The class *Assumption* represents the possible variabilities of assumptions resulted from the technique selected, such as data format, data type, splitting strategy, and others.

The *Test Design Generator* represents the second step and provides a class *Test Design, w*hich refers to variations identified in the feature diagram for the feature Test Design, as described in Section 2.2.1. The class *Test Design* is composed of the classes *Test Criteria* and *Test Strategy* which are the representation of the variation points identified for the Test Design step. The class *Test Criteria* expresses the variation points and variants related to the procedure to test the model's quality and validity. The class *Test Strategy* represents the variabilities of the data splitting process. It is composed of a class named *Data Splitting* which specifies the type, method, and techniques that may be applied in the splitting decisions.

The *Model Builder* represents the third step and the variabilities identified in the variability assessment, as described in Section 2.2.1. It provides the classes *Model* and *Model Tool*. The class *Model* represents variabilities related to the model, such as the parameters, score, conclusions, and description. *Model Tool* represents variabilities of the tool used for building the model, such as parameters. Not all tools use the same set of parameters to build a model.

Model Evaluator refers to forth step and the variabilities associated with it. The assessment step take into consideration strategy and criteria defined previously to provide the model rank. Classes *Assessment* and *Evaluation Result* represent the variabilities identified in the process of assessing a model. *Assessment* refers to the description of the result of the evaluation and the model rank, while *Evaluation Result* represents the criteria and the strategy for the model evaluation.

3.3 Evaluation of Automation Opportunities

The preliminary framework design described previously opens up several opportunities for automation in the data analysis modeling process. This sub-section, as illustrated in Figure 15, discusses these automation opportunities in each of the four CRISP-DM data analysis modeling tasks by contrasting the proposed framework design with the design solution provided in the SPSS Modeler - CRISP-DM [11].

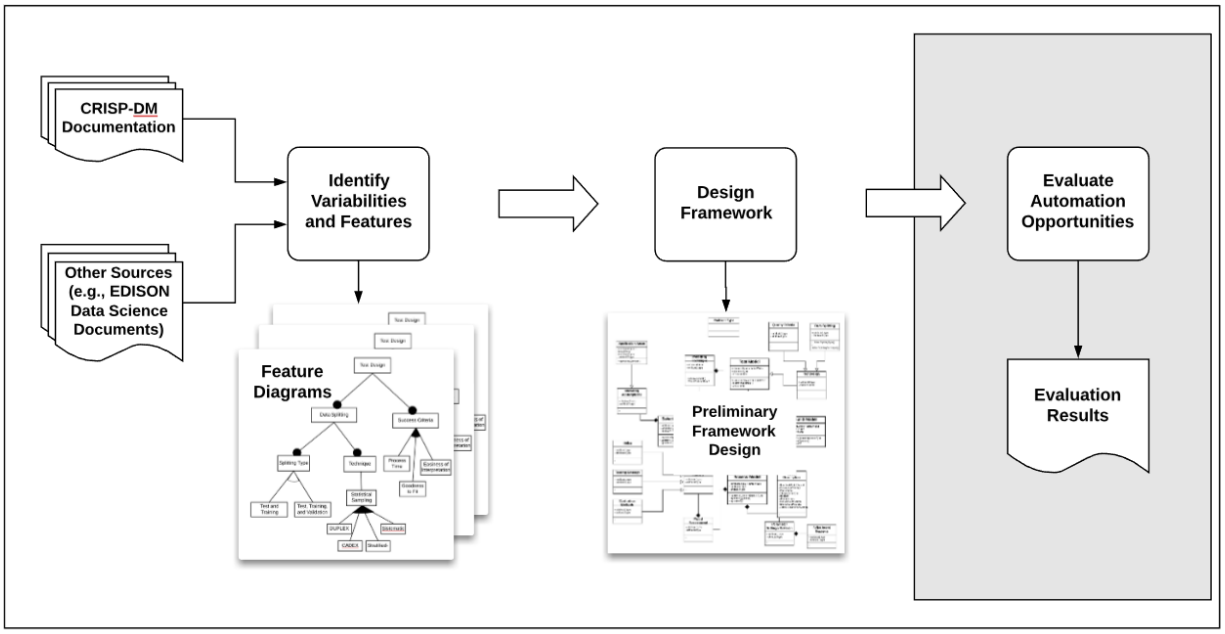

CRISP-DM Documentation
Other Sources (e.g., EDISON Data Science Documents)
Identify Variabilities and Features
Feature Diagrams
Design Framework
Preliminary Framework Design
Evaluate Automation Opportunities
Evaluation Results

Figure 15 - A variability-aware design approach: Evaluation of automation opportunities.

We have chosen to compare the framework design with the SPSS Modeler for CRISP-DM to assess automation opportunities for several reasons. First, SPSS Modeler is the only tool we have found that aims at automating the CRISP-DM data analysis process. Second, this tool has been used in many applications throughout the years, including applications in domains such as finance, marketing and smart cities. Third, although we did not have access to SPSS Modeler CRISP-DM requirements document, we could access a significant amount of information provided in the online user manual.

This section addresses the third research question: What are the opportunities for automation of the CRISP-DM data analysis modeling phase that go beyond the support provided by existing tools?

Overall, based on the variabilities that can be captured by our preliminary framework design, there are many opportunities for automating the CRISP-DM data analysis modeling process that go beyond the support provided by the existing SPSS Modeler for CRISP-DM. These opportunities for automation, which can significantly increase the level of automation of the CRISP-DM modeling process, include:

(i) The ability to support a more comprehensive set of modeling techniques and algorithms.
(ii) Mechanisms to record the assumptions required by each technique and algorithms in a more detailed way.
(iii) The ability to record the types of the data used by the models as input and output and the association with the algorithms that used them.
(iv) The ability to record and support data splitting choices, a more comprehensive set of quality criteria and success criteria;
(v) Mechanisms to support the generation of initial parameter settings.
(vi) The ability to record the rationale for parameter value choices.
(vii) Additional mechanisms for supporting interoperability in terms of the methods and input and output results.

(viii) The ability to record the resulting model characteristics, parameter settings, data quality issues, and provenance related to the data analysis method execution (e.g., what, who, when).

(ix) Mechanism to rank the models.

(x) Mechanisms for generating revised parameter settings.

The following sections provide the comparison between SPSS Modeler CRISP-DM and the preliminary framework design in terms of variability support for each one of the four generic tasks in the modeling phase. To represent the extent to which a specific variability has automated support, we use the symbol "+" ("-") to indicate that there is (not) automated support for the specific variability, and the symbol "+/-" to indicate that there is only partial automated support for it. The comparison was performed only by myself, and, of course, is subject of my own interpretation of the SPSS Modeler CRISP-DM tool documentation.

3.3.1 Select Modeling Technique

The Select Modeling Technique phase is divided into two subtasks: Modeling Technique and Modeling Assumptions. For **Modeling Technique**, Table 3 summarizes the comparison of the variability support provided by our framework design with the automation features provided in the SPSS Modeler CRISP-DM [11].

Automation Support Comparison: SELECTING MODELING TECHNIQUES		
Specific Variability	**SPSS Modeler - CRISP-DM**	**Framework Design**
Modeling Technique		
Data Analysis modeling approach	-	+
Modeling technique type	-	+
Model type	+/-	+
Algorithm type	+/-	+
Modeling Assumptions		
Data-related assumptions	+/-	+
Applications areas	+/-	+
Data used by the models	-	+

Table 3 - Comparison between SPSS Modeler CRISP-DM and framework design in the Selecting Modeling Techniques task.

There are several opportunities for automation in this task. As presented in Table 3, the SPSS Modeler for CRISP-DM tool does not cover all the automation possibilities that can be supported by our preliminary framework design. The tool does not support many opportunities for automation, including:

(i) The ability to support a more comprehensive set of modeling techniques, categorized by approach, technique type, model type, and algorithms.

(ii) Mechanisms to record the assumptions required by each technique and algorithms in a more detailed way. The set of a variety of types of data-related assumptions (e.g., specific modeling requirements, strategies for data splitting, data manipulations needed to meet the model requirements) can be useful for non-expertise users.

(iii) The ability to record the types of the data used by the models as input and output and the association with the algorithms that use them.

(iv) The record of the data used by the models.

(v) The ability to associate fields of application with techniques and assumptions.

3.3.2 Generate Test Design

The Generate Test Design phase is structured in only one subtask: Test design, and Table 4 summarizes the findings related to the comparison of our design solution for this subtask with the SPSS Modeler CRISP-DM tool.

Automation Support Comparison: GENERATE TEST DESIGN		
Specific Variability	**SPSS Modeler CRISP-DM**	**Framework Design**
Test design		
Data splitting choices	-	+
Data splitting techniques	-	+
Quality criteria	+/-	+
Success criteria	-	+
Strategy to stop	-	+

Table 4 - Comparison between SPSS Modeler CRISP-DM and framework design in the Generate Test Design task.

As presented in Table 4, the SPSS Modeler CRISP-DM tool does not support all possible variabilities that can be supported by our preliminary design approach, such as:

(i) The ability to encode data splitting techniques and record data splitting choices.

(ii) The ability to encode and record a more comprehensive set of quality and success criteria.

(iii) The ability to encode strategies to stop.

(iv) The possibility of evaluating and comparing results of every model running.

3.3.3 Build Model

The Build Model phase is divided into three subtasks: Parameter settings, Models, and Model description. Table 5 summarizes findings related to the comparison of our design solution in this subtask with the SPSS Modeler CRISP-DM tool.

Automation Support Comparison: BUILD MODEL		
Specific Variability	**SPSS Modeler CRISP-DM**	**Framework Design**
Parameter settings		
Setting parameter	+/-	+
Rationale for parameter value choices	+	+
Models		
Execution mode	-	+
Post-processing results	+/-	+
Preparation for interchangeable use	+/-	+
Model description		
Recorded description type	+/-	+

Table 5 - Comparison between SPSS Modeler CRISP-DM and framework design in the Build Model task.

As presented in Table 5, the SPSS Modeler CRISP-DM tool does not support variabilities such as:

(i) The generation of initial parameter settings.

(ii) The provision of additional mechanisms for supporting interoperability in terms of the methods and input and output results that can be used by other tools.

(iii) The ability to record the resulting model characteristics, parameter settings, data quality issues.

(iv) The ability to record provenance related to the data analysis method execution (e.g., what, who, when).

3.3.4 Assess Model

The Assess Model phase is divided into two subtasks: Model Assessment and Revised parameter settings. Table 6 summarizes findings related to the comparison of the design solution for this subtask with the SPSS CRISP-DM support tool.

Automation Support Comparison: ASSESS MODEL		
Specific Variability	**SPSS Modeler CRISP-DM**	**Framework Design**
Model assessment		
Evaluation criteria	+/-	+
Results evaluation	+/-	+
Results comments	-	+
Comparison of evaluation results	-	+
Model quality rank	+/-	+
Selected best models	+/-	+
Revised parameter settings		
Parameter adjustments	-	+
Adjustments reasons	-	+

Table 6: Comparison between SPSS Modeler – CRISP-DM and framework design in the Assess Model task.

As presented in Table 6, the SPSS Modeler CRISP-DM tool does not support variabilities such as:

(i) The ability to record detailed evaluation criteria, in terms of the rules, testing strategy, and evaluation methods.

(ii) The provision of several methods to assess the model according to recorded description type (e.g., provenance, parameter settings).

(iii) The ability to record task result summary and key steps of the evaluation process.

(iv) The ability to record the comments from data and domain experts.

(v) The ability to rank model quality.

(vi) The provision of additional techniques to support the selection of the best model.

(vii) The ability to support decisions on parameter revision and the ideal number of iterations.

3.4 Discussion

This section discusses the limitations of the proposed variability–aware design approach to the data analysis modeling process.

First, there are many different methods to support the variability of software systems, and our proposal relied on an informal approach based on UML, in which the verbs and nouns of requirement documents are analyzed in terms of possible variabilities. For example, when the noun “method” is found, potentially this method can be a variation point, and multiple methods can be selected. Although the approach to derive classes and variabilities is manual, there are different methods and tools proposed in the literature that could be used to support the derivation of UML structural diagrams, from which the variabilities can be analyzed. We have not used these tools to support the automatic derivation of UML class diagrams because: (i) our study is exploratory in nature; (ii) the tools still relies heavily on user input and provide an explosive number of classes; and (iii) the documentation used as the basis for the analysis is not a detailed software requirements document, but we had to rely on descriptions provided by user manuals.

Second, preliminary framework design presented in this chapter only depicts the high level abstractions and relationships that capture the main variabilities in the CRISP-DM data analysis modeling process. We understand that such a preliminary solution depicts the richness of the variabilities present in this data analysis modeling process and any refinement of this design towards implementation would need specific domain-based information about intended applications. In addition, we do not claim the proposed framework design is complete because we tried to be general in the solution, but we believe that more complete solutions can be provided when particular domain-specific application scenarios are investigated.

Third, it is difficult to find information about some of the specific aspects of tool support when it comes to proprietary software such as SPSS CRISP-DM. This made the comparison of our framework design with this proprietary tool, in some cases, hard to make.

Chapter 4
Conclusions and Future Work

4.1 Conclusions

This thesis proposes a variability-aware design approach to the data analysis modeling process. The approach involves (i) the assessment of the variabilities inherent in the CRISP-DM data analysis modeling phase and the provision of feature models that represent these variabilities; (ii) the definition of a preliminary framework design that captures the identified variabilities; and (iii) evaluation of the developed framework design in terms of the possibilities for automation.

The proposed approach helps to advance the state of the art by providing potential design enhancements to existing solutions and indicating novel ways to automate the data analysis modeling process, as described in Section 3.3. These results are beneficial both to designers and practitioners who are involved in the design and implementation of processes to support the data analysis modeling phase.

4.2 Future work

Future work related to this thesis may involve: (i) the refinement and implementation of the proposed framework design; (ii) the development of specific case studies in particular domains; (iii) the extension of the approach to other data analysis processes and other phases; (iv) the use of (knowledge) databases to capture data relevant to the process; and (v) the provision of automated learning capabilities that can provide user guidance.

The proposed framework design can be refined and implemented in the future. This work will involve the refinement of the framework class diagram and the definition of the framework behavior using other UML diagrams such as the UML sequence and collaboration diagrams. The implementation can be based on existing object-oriented languages such as Java and C++, which can be combined with libraries and tools to support the execution of data analysis algorithms.

Case studies can be developed to address a subset of the methods in specific domains. It would be interesting to investigate applications in areas such as smart cities and environmental applications. In the case of smart cities, the design solutions could, in principle, provide more flexible tools to support the data analysis modeling process in order to investigate patterns and trends related, for example, to social and economic development factors. In terms of environmental applications, the proposed preliminary framework could provide a starting point for the design and implementation of decision support systems for surface water management that help detect floods, droughts or other extreme weather events.

The proposed variability-aware approach can also be extended and applied to other data analysis processes such as SEMMA and other phases of the data analysis process, which include business understanding, data understanding, data preparation, evaluation and deployment. SEMMA addresses a subset of the phases covered by CRISP-DM, namely data understanding, data preparation, modeling and evaluation, but has different sub-phases, which aim to sample, explore, modify, model and assess.

The introduction of knowledge and data bases in the framework design can help to capture relevant information associated not only with the data analysis modeling process but to other tasks of the CRISP-DM approach. Several forms of provenance, for example, can be captured and stored, including contextual information about what, who, when, and how. This information may involve the data used, the algorithms used, as well as their input and output for each session. Data analysis processes can also be captured and stored and can be reused in future execution scenarios.

Finally, the extension of the proposed approach and its implementation with automated learning techniques based on data history can help to support user guidance. This could lead to potential solutions that involve recommendation techniques able to provide suggestions, for example, of the specific data and methods to be used.

References

[1] S. Sumathi and S.N. Sivanandam, "Introduction to Data Mining Principles," in Introduction to Data Mining and its Applications, S. Sumathi and S.N. Sivanandam, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1-20.

[2] S. Mitra, S.K. Pal and P. Mitra, "Data Mining in Soft Computing Framework: a Survey," IEEE Transactions on Neural Networks, vol. 13, pp. 3-14, 2002.

[3] S. Tyagi, "Using Data Analytics for Greater Profits," Journal of Business Strategy, vol. 24, pp. 12-14, 2003.

[4] A. Bănărescu, "Detecting and Preventing Fraud with Data Analytics," Procedia Economics and Finance, vol. 32, pp. 1827-1836, 2015.

[5] T.A. Runkler, "Introduction," in Data Analytics: Models and Algorithms for Intelligent Data Analysis, T.A. Runkler, Wiesbaden: Springer Fachmedien Wiesbaden, 2016, pp. 1-3.

[6] A. Famili, W. Shen, R. Weber and E. Simoudis, "Data Preprocessing and Intelligent Data Analysis," Intelligent Data Analysis, vol. 1, pp. 3-23, 1997.

[7] A. Holzinger, M. Dehmer and I. Jurisica, "Knowledge Discovery and Interactive Data Mining in Bioinformatics--State-of-the-Art, Future Challenges and Research Directions," BMC Bioinformatics, vol. 15, 2014.

[8] Z. H. Zhou, N. V. Chawla, Y. Jin and G. J. Williams, "Big Data Opportunities and Challenges: Discussions from Data Analytics Perspectives," IEEE Computational Intelligence Magazine, vol. 9, pp. 62-74, 2014.

[9] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer and R. Wirth, "CRISP-DM 1.0: Step-by-Step data mining Guide," 2000.

[10] SAS Institute Inc., "Data Mining and the Case for Sampling. Solving Business Problems UsingSAS® Enterprise Miner™ Software," pp. 36, 1998.

[11] IBM Software Group, "IBM SPSS Modeler CRISP-DM Guide," 2015.

[12] M. Kantardzic, Data Mining: Concepts, Models, Methods, and Algorithms, Hoboken, NJ: Wiley-IEEE Press, 2011, .

[13] R. Capilla and J. Bosch, "Software Variability and Design Decisions," in Systems and Software Variability Management: Concepts, Tools and Experiences, R. Capilla, J.

Bosch and K. Kang, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 287-292.

[14] M. Svahnberg, J. van Gurp and J. Bosch, "A Taxonomy of Variability Realization Techniques," Software: Practice and Experience, vol. 35, pp. 705-754, 2005.

[15] A. K. Thurimella, B. Bruegge and O. Creighton, "Identifying and Exploiting the Similarities between Rationale Management and Variability Management," in 12th International Software Product Line Conference, pp. 99-108, 2008.

[16] T. Forster, D. Muthig and D. Tech, "Understanding Decision Models – Visualization and Complexity Reduction of Software Variability," in Second International Workshop on Variability Modeling of Software-Intensive Systems, pp. 111-119, 2008.

[17] M. Galster, D. Weyns, D. Tofan, B. Michalik and P. Avgeriou, "Variability in Software Systems - A Systematic Literature Review," IEEE Transactions on Software Engineering, vol. 40, pp. 282-306, 2014.

[18] M. Galster and P. Avgeriou, "A Variability Viewpoint for Enterprise Software Systems," pp. 267-271, Aug 2012.

[19] F. Bachmann and P. C Clements, Variability in Software Product Lines, 2005, pp. 3.

[20] H. Ye and H. Liu, "Approach to Modelling Feature Variability and Dependencies in Software Product Lines," IEE Proceedings - Software, vol. 152, pp. 101, 2005.

[21] K. Lauenroth and K. Pohl, "Principles of Variability," in Software Product Line Engineering: Foundations, Principles, and Techniques, K. Pohl, G. Böckle and F. van der Linden, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 57-88.

[22] Yukyong Kim and Kyung-Goo Doh, "Adaptable Web Services Modeling Using Variability Analysis," in Third International Conference on Convergence and Hybrid Information Technology, pp. 700-705, 2008.

[23] J. Van Gurp, J. Bosch and M. Svahnberg, "On the Notion of Variability in Software Product Lines," in IEEE/IFIP Conference on Software Architecture, pp. 45-54, 2001.

[24] K.C. Kang and H. Lee, "Variability Modeling," in Systems and Software Variability Management: Concepts, Tools and Experiences, R. Capilla., J. Bosch and K.C. Kang, Springer Berlin Heidelberg, 2013, pp. 25-42.

[25] J. Bosch, "Software Variability Management," in Proceedings of International Conference on Software Engineering, pp. 720-721, 2004.

[26] M. Sinnema and S. Deelstra, "Classifying Variability Modeling Techniques," Information and Software Technology, vol. 49, pp. 717-739, 2007.

[27] H. Gomaa, Designing Software Product Lines with UML: from Use Cases to Pattern-based Software Architectures, Boston: Addison-Wesley, 2005, .

[28] K. Czarnecki, P. Grünbacher, R. Rabiser, K. Schmid and A. Wąsowski, "Cool Features and Tough Decisions: A Comparison of Variability Modeling Approaches," in Proceedings of the Sixth International Workshop on Variability Modeling of software-intensive systems, pp. 173-182, 2012.

[29] S. Srinivasan and J. Vergo, "Object Oriented Reuse: Experience in Developing a Framework for Speech Recognition Applications," in Proceedings of the 20th International Conference on Software Engineering, pp. 322-330, 1998.

[30] J. Bosch, P. Molin, M. Mattsson and P. Bengtsson, "Object-Oriented Frameworks - Problems and Experiences," ACM Computing Surveys, vol. 32, 1997.

[31] R.E. Johnson and B. Foote, "Designing Reusable Classes," Journal of Object-Oriented Programming, vol. 1, pp. 30, 35, 1988.

[32] M.E. Caspersen and H.B. Christensen, "Frameworks in Teaching," in Reflections on the Teaching of Programming: Methods and Implementations, J. Bennedsen, M.E. Caspersen and M. Kölling, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 190-205.

[33] J. Rumbaugh, I. Jacobson and G. Booch, The Unified Modeling Language Reference Manual, Boston: Addison-Wesley, 2005, .

[34] R.E. Johnson, "Frameworks - Components Patterns," Communications of the ACM, vol. 40, pp. 39-42, 1997.

[35] Y. Zhu and S.M. Cochoff, "An Object-oriented Framework for Medical Image Registration, Fusion, and Visualization," Computer Methods and Programs in Biomedicine, vol. 82, pp. 258-267, 2006.

[36] T.C. Oliveira, P. Alencar and D. Cowan, "ReuseTool—An Extensible Tool Support for Object-oriented Framework Reuse," Journal of Systems and Software, vol. 84, pp. 2234-2252, 2011.

[37] S. Srinivasan, "Design Patterns in Object-Oriented Frameworks," Computer, vol. 32, pp. 24-32, 1999.

[38] M. Fayad and D.C. Schmidt, "Object-Oriented Application Frameworks," Communications of the ACM, pp. 32-38, 1997.

[39] C. Kobryn, "Modeling Components and Frameworks with UML," Communications of the ACM, vol. 43, pp. 31-38, 2000.

[40] M. Fayad, D. Hamu and D. Brugali, "Enterprise Frameworks Characteristics, Criteria, and Challenges," Communications of the ACM, vol. 43, pp. 39-46, 2000.

[41] G. Larsen, "Component-based Enterprise Frameworks," Communications of the ACM, vol. 43, pp. 24-26, 2000.

[42] G. Booch, "Object-oriented Development," IEEE Transactions on Software Engineering, vol. 12, pp. 211-221, 1986.

[43] G. Mariscal, Ó Marbán and C. Fernández, "A Survey of Data Mining and Knowledge Discovery Process Models and Methodologies," The Knowledge Engineering Review, vol. 25, pp. 137-166, 2010.

[44] R. Nisbet, G. Miner and K. Yale, "Chapter 3 - The Data Mining and Predictive Analytic Process," Handbook of Statistical Analysis and Data Mining Applications, pp. 39-54, 2018.

[45] R. Nisbet, J.F. Elder and G. Miner, Handbook of Statistical Analysis and Data Mining Applications, Amsterdam: Elsevier Inc., 2009, .

[46] C. Catley, K. Smith, C. McGregor and M. Tracy, "Extending CRISP-DM to Incorporate Temporal Data Mining of Multidimensional Medical Data Streams: A Neonatal Intensive Care Unit Case Study," 22nd IEEE International Symposium on Computer-Based Medical Systems, pp. 1-5, 2009.

[47] F. Martínez-Plumed, L. Contreras-Ochando, C. Ferri, P. Flach, J. Hernández-Orallo, M. Kull, N. Lachiche and M.J. Ramírez-Quintana, "CASP-DM: Context Aware Standard Process for Data Mining," CoRR, 2017.

[48] O. Niaksu, "CRISP Data Mining Methodology Extension for Medical Domain," Baltic Journal of Modern Computing, vol. 3, pp. 92, 2015.

[49] J. Venter, A. de Waal and C. Willers, "Specializing CRISP-DM for Evidence Mining," in Advances in Digital Forensics III, New York, NY: Springer New York, 2007, pp. 303-315.

[50] M. Galster, D. Weyns, M. Goedicke, U. Zdun, J. Cunha and J. Chavarriaga, "Variability and Complexity in Software Design - Towards Quality through Modeling and Testing," ACM SIGSOFT Software Engineering Notes, vol. 42, pp. 35-37, 2018.

[51] L. Mich, "NL-OOPS: from Natural Language to Object Oriented Requirements using the Natural Language Processing System LOLITA," Natural Language Engineering, vol. 2, pp. 161-187, 1996.

[52] T. Berger, D. Lettner, J. Rubin, P. Grünbacher, A. Silva, M. Becker, M. Chechik and K. Czarnecki, "What is a Feature?" pp. 16-25, Jul 20, 2015.

[53] K.C. Kang, S.G. Cohen, J.A. Hess, W.E. Novak and A.S. Peterson, "Feature-Oriented Domain Analysis (FODA) Feasibility Study," 1990.

[54] K. Czarnecki, S. Helsen and U. Eisenecker, "Staged Configuration Using Feature Models," in Software Product Lines, pp. 266-283, 2004.

[55] T. Thum, D. Batory and C. Kastner, "Reasoning about Edits to Feature Models," pp. 254-264, May 16, 2009.

[56] K. Czarnecki, "Overview of Generative Software Development," in Unconventional Programming Paradigms, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 326-341.

[57] M. Riebisch, D. Streitferdt and I. Pashov, "Modeling Variability for Object-Oriented Product Lines," in Object-Oriented Technology. ECOOP 2003 Workshop Reader, pp. 165-178, 2004.

[58] D. Batory, "Feature Models, Grammars, and Propositional Formulas," in Software Product Lines, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 7-20.

[59] M. Mendonca, A. Wąsowski and K. Czarnecki, "SAT-based Analysis of Feature Models is Easy," pp. 231-240, 2009.

[60] I. Pashov and M. Riebisch, "Using feature modeling for program comprehension and software architecture recovery," pp. 406-417, 2004.

[61] D. Deeptimahanti and M. Babar, "An Automated Tool for Generating UML Models from Natural Language Requirements," in IEEE/ACM International Conference on Automated Software Engineering, pp. 680-682, 2009.

[62] I. Song, K. Yano, J. Trujillo and S. Lujan-Mora, "A Taxonomic Class Modeling Methodology for Object-Oriented Analysis," in Information Modeling Methods and Methodologies: Advanced Topics in Database Research, J. Krogstie, T. Halpin and K. Siau, Idea Publishing Group Global, 2005, pp. 216-240.

[63] M. Sinnema, S. Deelstra, J. Nijhuis, J. Bosch and R.L. Nord, "COVAMOF: A Framework for Modeling Variability in Software Product Families," pp. 197-213, 2004.

[64] M. Galster, U. Zdun, D. Weyns, R. Rabiser, B. Zhang, M. Goedicke and G. Perrouin, "Variability and Complexity in Software Design : Towards a Research Agenda," ACM SIGSOFT Software Engineering Notes, 2016.

[65] J. Horcas, M. Pinto and L. Fuentes, "Variability Models for Generating Efficient Configurations of Functional Quality Attributes," Information and Software Technology, vol. 95, pp. 147-164, 2018.

[66] G.H. Alférez, V. Pelechano, R. Mazo, C. Salinesi and D. Diaz, "Dynamic Adaptation of Service Compositions with Variability Models," Journal of Systems and Software, vol. 91, pp. 24-47, 2014.

[67] I. Reinhartz-Berger and A. Sturm, "Comprehensibility of UML-based Software Product Line Specifications," Empirical Software Engineering, vol. 19, pp. 678-713, 2014.

[68] C. Ayora, V. Torres, B. Weber, M. Reichert and V. Pelechano, "VIVACE: A framework for the Systematic Evaluation of Variability Support in Process-aware Information Systems," Information and Software Technology, vol. 57, pp. 248-276, 2015.

[69] M. Koning, C. Sun, M. Sinnema and P. Avgeriou, "VxBPEL: Supporting Variability for Web Services in BPEL," Information and Software Technology, vol. 51, pp. 258-269, 2009.

[70] F. Berzal, J. Cubero and A. Jimenez, "The Design and Use of the TMiner Component-based Data Mining Framework," Expert Systems with Applications, vol. 36, pp. 7882-7887, 2009.

[71] S. Wang and C. Eick, "A Data Mining Framework for Environmental and Geo-spatial Data Analysis," International Journal of Data Science and Analytics, vol. 5, pp. 83-98, 2018.

[72] M. Khanbabaei, F.M. Sobhani, M. Alborzi and R. Radfar, "Developing an Integrated Framework for Using Data Mining Techniques and Ontology Concepts for Process Improvement," Journal of Systems and Software, vol. 137, pp. 78-95, 2018.

[73] Y. Zou, Z. Xiao, L. Zhang, E. Zio, J. Liu and H. Jia, "A Data Mining Framework within the Chinese NPPs Operating Experience Feedback System for Identifying Intrinsic Correlations among Human Factors," Annals of Nuclear Energy, vol. 116, pp. 163-170, 2018.

[74] R. Brun and F. Rademakers, "ROOT — An Object Oriented Data Analysis Framework," Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, vol. 389, pp. 81-86, 1997.

[75] B. Ulfenborg, A. Karlsson, M. Riveiro, C. Améen, K. Åkesson, C.X. Andersson, P. Sartipy and J. Synnergren, "A Data Analysis Framework for Biomedical Big data: Application on Mesoderm Differentiation of Human Pluripotent Stem Cells," Plos One, vol. 12, 2017.

[76] M. Krätzig, "A Software Framework for Data Analysis," Computational Statistics & Data Analysis, vol. 52, pp. 618-634, 2007.

[77] EDISON Project, "EDISON Data Science Framework (EDSF)," Http://Edison-Project.Eu/Edison/Edison-Data-Science-Framework-Edsf, 2017.

[78] Y. Demchenko, A. Belloum, W. Los, T. Wiktorski, A. Manieri, H. Brocks, J. Becker, D. Heutelbeck, M. Hemmje and S. Brewer, "EDISON Data Science Framework: A Foundation for Building Data Science Profession for Research and Industry," pp. 620-626, 2016.

[79] P. Kashyap, "Machine Learning Algorithms and Their Relationship with Modern Technologies," in Machine Learning for Decision Makers, Berkeley, CA: Apress, 2017, pp. 91-136.

[80] P.d. Saqui-Sannes and L. Apvrille, "Making Modeling Assumptions an Explicit Part of Real-Time Systems Models," in 8th European Congress on Embedded Real Time Software and Systems, 2016.

[81] C.C. Aggarwal and K.R. Chandan, Data Clustering: Algorithms and Applications, Boca Raton: Chapman & Hall/CRC, 2013, pp. 652.

[82] D. Kitakoshi, S. Wakasaki and M. Suzuki, "A Probabilistic Reasoning Algorithm for Bayesian Networks by Simplifying their Structures," pp. 336-341, 2011.

[83] S. Anand, P. Padmanabham, A. Govardhan and R.H. Kulkarni, "An Extensive Review on Data Mining Methods and Clustering Models for Intelligent Transportation System," Journal of Intelligent Systems, vol. 27, pp. 263-273, Mar 28,. 2018.

[84] M. Vazirgiannis and M. Halkidi, "Quality Assessment Approaches in Data Mining," in Data Mining and Knowledge Discovery Handbook, Boston, MA: Springer US, 2005, pp. 661-696.

[85] Z. Reitermanova, "Data Splitting," WDS'10 Proceedings of Contributed Papers, pp. 31-36, 2010.

Appendix A
CRISP-DM Variability Assessment

In this appendix, an assessment of the variabilities inherent in the CRISP-DM data analysis modeling phase is presented, in which variabilities are identified for the four generic tasks of the CRISP-DM modeling phase, namely Select Modeling Technique, Generate Test Design, Build Model and Assess Model.

Task 1 – Select Modeling Technique

This task is divided into two subtasks: (1.1) Modeling Techniques; and (1.2) Modeling Assumptions.

(1.1) Modeling Techniques: The variations in this subtask involve the following categories: data analysis modeling approach; modeling technique type; the model type, and the algorithm type. The structure for the variations in these categories can be represented as follows:

- (i) Statistical methods
  - (i.1) Basic probability and statistics
    - (i.1.1) Basic statistics
      - (i.1.1.1) Mean
      - (i.1.1.2) Variance
  - (i.2) Statistical paradigms
    - (i.2.1) Regression
    - (i.2.2) Time series
    - (i.2.3) Dimensionality
    - (i.2.4) Clusters

(i.2.5) Probabilistic representation/reasoning

(i.2.6) Causal networks

(i.2.6.1) Belief Propagation

(i.2.6.2) Sampling

(i.2.7) Bayesian analysis

(i.2.8) Markov nets

(i.3) Exploratory data analysis

(i.3.1) Stem and leaf plot

(i.3.2) Boxplot

(i.3.3) Median

(i.3.4) Histograms

(i.3.5) Scatter plots

(i.3.6) DEX Scatter Plot

(i.3.7) DEX Mean Plot

(i.3.8) Transformations

(i.3.9) Quantile-Quantile plot - QQ-Plot

(i.3.10) Interquartile range

(i.4) Confirmatory data analysis

(i.5) Frequentist Bayesian statistics

(i.6) Quantitative analytics

(i.7) Performance analysis

(i.8) Markov models and Markov networks

(ii) Machine learning

- (ii.1) Supervised machine learning
  - (ii.1.1) Decision trees
    - (ii.1.1.1) ID3
    - (ii.1.1.2) SLIQ
    - (ii.1.1.3) SPRINT
    - (ii.1.1.4) C4.5
    - (ii.1.1.5) C5.0
  - (ii.1.2) Random forest
  - (ii.1.3) Naïve Bayes
  - (ii.1.4) Classification
    - (ii.1.4.1) Decision tree
    - (ii.1.4.2) Logistic regression
    - (ii.1.4.3) Naïve Bayes
    - (ii.1.4.4) Discriminant Analysis
    - (ii.1.4.5) Support Vector Machines – SVM
  - (ii.1.5) Ordinary least square regression
  - (ii.1.6) Logistic regression
  - (ii.1.7) Locally Weighted Regression
  - (ii.1.8) Q-Learning Algorithm
  - (ii.1.9) Back Propagation Algorithm
  - (ii.1.10) Neural networks

(ii.1.10.1) Backpropagation

(ii.1.10.2) Radial Base Function (RBF)

(ii.1.11) Support vector machines (SVMs);

(ii.1.12) Ensemble methods

(ii.1.12.1) Bagging

(ii.1.12.2) Bootstrapping

(ii.2) Unsupervised machine learning

(ii.2.1) Clustering based

(ii.2.1.1) K-means

(ii.2.1.2) K-Medoids

(ii.2.1.3) Partitioning Around Medoids (PAM)

(ii.2.1.4) Clustering Large Applications (CLARA)

(ii.2.1.5) Single-linkage

(ii.2.1.6) Mixture of Gaussians

(ii.2.1.7) Fuzzy C-Means

(ii.2.2) Density Based

(ii.2.2.1) DBSCAN

(ii.2.2.2) Ordering Points to Identify the Clustering Structure (OPTICS)

(ii.2.2.3) Density Based Clustering (DENCLUE)

(ii.2.3) Grid Based

(ii.2.3.1) Statistical Information Grid (STING)

- (ii.2.4) Model based
    - (ii.2.4.1) COWEB
    - (ii.2.4.2) CLASSIT
- (ii.2.5) Hierarchical
    - (ii.2.5.1) Agglomerative Nesting (AGNES)
    - (ii.2.5.2) CHAMELEON
    - (ii.2.5.3) Clustering Using Representatives (CURE)
    - (ii.2.5.4) Divisive Analysis (DIANA)
- (ii.2.6) Principal Component Analysis (PCA)
- (ii.2.7) Singular Value Decomposition (SVD)
    - (ii.2.7.1) Principal Direction Divisive Partitioning (PDDP)
- (ii.2.8) Independent Component Analysis (ICA)
- (ii.2.9) Markov-Modulated Poison Process (MMPP)
- (ii.2.10) LOF
- (ii.2.11) Exponential Smoothing

(ii.3) Reinforcement learning

- (ii.3.1) Q-Learning
- (ii.3.2) Temporal Difference Learning (TD-Learning)
- (ii.3.3) State-Action-Reward-State-Action (SARSA)
- (ii.3.4) Deep Deterministic Policy Gradient (DDPG)
- (ii.3.5) Deep Q Network (DQN)
- (ii.3.6) Markov decision Process

        (ii.3.7) Genetic algorithms

    (ii.4) Classification methods

        (ii.4.1) Classification-Based Association (CBA)

        (ii.4.2) Classification-Based on Multiple Association Rules (CMAR)

        (ii.4.3) Classification Association Rule Mining (CARM)

        (ii.4.4) Classification-Based on Predictive Association Rules (CPAR)

    (ii.5) Multi-task learning

    (ii.6) Statistical paradigms

    (ii.7) Probabilistic representation/reasoning

    (ii.8) Frequentist Bayesian statistics

    (ii.9) Quantitative analytics

    (ii.10) Performance analysis

    (ii.11) Markov models and Markov networks

(iii) Data mining

    (iii.1) Performance analysis

    (iii.2) Anomaly detection

    (iii.3) Time series analysis

        (iii.3.1) K-Means clustering

    (iii.4) Feature selection

        (iii.4.1) Apriori algorithm

    (iii.5) Graph data analytics

(iv) Predictive analytics

- (iv.1) Machine learning for predictive analytics
- (iv.2) Regression and multi analysis
- (iv.3) Generalized linear models
- (iv.4) Time series analysis and forecasting

(v) Computational modeling simulation/optimization

- (v.1) Modeling and simulation techniques
- (v.2) Operations research and optimization techniques
- (v.3) Large-scale modeling and simulation system techniques
- (v.4) Network optimization methods
- (v.5) Risk simulation and queuing techniques

(vi) Domain-specific data analytics methods

- (vi.1) Text data mining
  - (vi.1.1) Text analytics
    - (vi.1.1.1) Statistical analysis
    - (vi.1.1.2) Linguistic analysis
    - (vi.1.1.3) Structural techniques
  - (vi.1.2) Natural language processing (NL)
  - (vi.1.3) Predictive models for text
  - (vi.1.4) Retrieving and clustering of documents
  - (vi.1.5) Information extraction
  - (vi.1.6) Sentiment analysis
- (vi.2) Business analytics methods

(vii) Problem type

- (vii.1) Description and summarization
- (vii.2) Segmentation
- (vii.3) Concept description
- (vii.4) Classification
- (vii.5) Prediction
- (vii.6) Dependency analysis

(1.2) Modeling Assumptions: The categories of variations for modeling assumptions are represented in the following structure: data-related assumptions, application areas, and data used by the models.

(i) Data-related assumptions:

- (i.1) Basic data assumptions
  - (i.1.1) Data format
  - (i.1.2) Data quality and constraints
    - (i.1.2.1) uniform distributions
    - (i.1.2.2) missing values allowed
  - (i.1.3) Data types
    - (i.1.3.1) Nominal
    - (i.1.3.2) Ordinal
    - (i.1.3.3) Interval
    - (i.1.3.4) Ratio
- (i.2) Data mining goals

- (i.2.1) Gain insight
- (i.2.2) Produce a score
- (i.2.3) Patterns
- (i.2.4) Clusters
- (i.2.5) Versatile models

(i.3) Specific modeling requirements

- (i.3.1) Particular data size or type
- (i.3.2) Easiness of printing results

(i.4) Data splitting into test/training/validation sets

- (i.4.1) No splitting
- (i.4.2) Test/training/validation splitting strategies

(i.5) Data availability to produce reliable results by the modeling technique

- (i.5.1) not enough data
- (i.5.2) enough data available

(i.6) Data quality level required by the modeling technique

- (i.6.1) specific data quality level required
- (i.6.2) not required

(i.7) Proper data type required by the modeling technique

- (i.7.1) Appropriate
- (i.7.2) Non-appropriate data

(i.8) Data manipulations or transformations needed to meet the model requirements (i.8.1) Transform

(i.8.2) Convert

(i.8.3) Rebalance the data

(i.9) Known data types before execution

(i.9.1) Logistic regression and neural networks require known data types before execution

(i.9.2) Rebalance the data when predicting rule events

(i.10) Role that each data field plays in the model technique

(i.10.1) Target

(i.10.2) Input fields

(ii) Application areas

(ii.1) Advanced planning and scheduling (APS)

(ii.2) Quality improvement

(ii.3) Spatial data

(ii.4) Marketing

(ii.5) Defect analysis

(ii.6) Fault diagnosis

(iii) Data used by the models

(iii.1) Structured

(iii.2) Semi-structured

(iii.3) Unstructured

Task 2 – Generate Test Design

This task comprises a single subtask, namely (2.1) Test Design.

(2.1) Test Design: The variations for this task are grouped into the following task structure: data splitting, quality criteria, test design variations, success criteria, and strategy to stop.

- (i) Data splitting choices
  - (i.1) No splitting
  - (i.2) Test and Training
  - (i.3) Test, Training, and Validation
- (ii) Data splitting and sampling techniques
  - (ii.1) Cross validation
    - (ii.1.1) Hold-out
    - (ii.1.2) k-Fold
    - (ii.1.3) Bootstrapping
  - (ii.2) Statistical sampling
    - (ii.2.1) Simple random sampling (SRS)
    - (ii.2.2) Trial-and-error methods
    - (ii.2.3) Systematic sampling
    - (ii.2.4) Convenience sampling
    - (ii.2.5) CADEX, DUPLEX
    - (ii.2.6) Stratified sampling
- (iii) Quality criteria
  - (iii.1) Accuracy
  - (iii.2) Sensitivity
  - (iii.3) Specificity
  - (iii.4) Coverage
  - (iii.5) Support
  - (iii.6) Confidence
  - (iii.7) Leverage
  - (iii.8) Lift
  - (iii.9) Conviction
  - (iii.10) Receiver Operating Characteristics (ROC)

- (iii.11) Recall, precision and Measure–F
  - (iii.11.1) Recall
  - (iii.11.2) Precision
  - (iii.11.3) Measure–F
- (iii.12) Other Measures
  - (iii.12.1) Mean absolute error
  - (iii.12.2) Root mean square error
  - (iii.12.3) Relative absolute error
  - (iii.12.4) Root relative squared error

(iv) Success criteria

- (iv.1) Goodness of fit
- (iv.2) Ease of interpretation
- (iv.3) Ease of deployment
- (iv.4) Required processing time

(v) Strategy to stop

- (v.1) Manual
  - (v.1.1) # of iterations (Rerun)
- (v.2) Automated

Task 3 – Build Model

This task is divided into two subtasks: (3.1) Parameter Settings; (3.2) Models; and (3.3) Model Description.

(3.1) Parameter Settings: The variations in this task involve the following categories: setting parameters, rationale for parameter value choices.

(i) Setting parameters

- (i.1) Manual
- (i.2) Automated
  - (i.2.1) Default

(i.2.2) Parameter setting techniques

(ii) Rationale for parameter value choices

(ii.1) Manual

(ii.2) Automated

(3.2) Models: The categories of variations for this subtask are represented as follows: execution mode, post-processing results; and preparation for interchangeable use.

(i) Execution mode

(i.1) Sequential

(i.2) Parallel

(ii) Post-processing results

(ii.1) No post-processing

(ii.2) Post-processing procedures

(ii.2.1) Edit

(ii.2.2) Display

(iii) Preparation for interchangeable use

(iii.1) Predictive Model Markup Language (PMML)

(iii.2) Weka Attribute Relation File Format (ARFF)

(3.3) Model Description: The variations in this subtask involve only one criteria, recorded description type. The structure for the variations in this category can be represented as follows:

(i) Recorded description type

(i.1) Resultant model characteristics (e.g., neural network topology)

(i.2) Parameter settings

- (i.3) Provenance (e.g., what, who, when, where)
- (i.4) Data quality issues (e.g., high number of missing values)
- (i.5) Behavior (e.g., quality metrics such as accuracy, sensitivity, coverage)
- (i.6) Interpretation (e.g., in business terms)
- (i.7) Shortcomings (e.g., execution problems, processing time)
- (i.8) Derived conclusions (e.g., discovered insights, patterns, anomalies)

Task 4 – Assess Model

This task is divided into two subtasks: (4.1) Model Assessment; and (4.2) Revised Parameter Settings.

(4.1) Model Assessment: The variations in this task involve the following category structure: task summary; description type; and quality task results, model qualities, and quality rank.

- (i) Task summary
  - (i.1) Manual
  - (i.2) Automated
- (ii) Recorded description type
  - (ii.1) Deployment potential of the model
  - (ii.2) Optimal parameter settings
- (iii) Assessment according to recorded description type
  - (iii.1) Resultant model characteristics (e.g., neural network topology)
  - (iii.2) Parameter settings
  - (iii.3) Provenance (e.g., what, who, when, where)
  - (iii.4) Data quality issues (e.g., high number of missing values)
  - (iii.5) Behavior (e.g., success and quality metrics such as accuracy, sensitivity, coverage)
  - (iii.6) Interpretation (e.g., in business terms with respect to goals)
  - (iii.7) Shortcomings (e.g., execution problems, processing time)

        (iii.8) Derived conclusions (e.g., discovered insights, patterns, anomalies)
        (iii.9) Deployment potential of the model
        (iii.10) Optimal parameter settings
    (iv) Comparison of evaluation results
    (v) Model quality rank
        (v.1) Manual
        (v.2) Automated
    (vi) Selected best models

(4.2) Revised Parameter Settings

- (i) Decision on setting revised parameters
    - (i.1) No parameters revision
    - (i.2) Parameters revision
- (ii) Iteration definition
    - (ii.1) Manual
    - (ii.2) Automated
- (iii) Return point
    - (iii.1) Task 1: Select model technique
    - (iii.2) Task 3: Build model

Index

Concept feature, 29

CRISP-DM, 22

Cross-tree constraints, 30

Cross-tree relationship, 28

Data analysis modeling process, 61

Feature diagram, 28

Feature model, 28

Framework, 14

Mandatory Feature, 29

Object-oriented framework, 32

Optional Feature, 29

UML class diagrams, 30

Variability, 13

Variability model, 14